International Journal of Networking
ISSN: 2249-278X & E-ISSN: 2249-2798, Volume 2, Issue 1, 2012, pp.-40-43.
Available online at http://www.bioinfo.in/contents.php?id=108

PROCESSING OF CONCURRENT TRANSACTIONS IN A PEER TO PEER DATABASE NETWORK SYSTEM

MISHRA A.K.*, UMARE T.V., ATAL R.S. AND MAKHIJA R.R.

Department of Computer Science & Engineering, Amravati University, Yavatmal, Maharashtra, India.
*Corresponding Author: Email- aash9969@rediffmail.com

Received: February 21, 2012; Accepted: March 15, 2012

Abstract- In this paper a concurrent execution of a transaction in a peer to peer database network is investigated. A peer to peer database network is nothing but a collection of peers connected in a network which shares the databases for the sake of convenience with an interoperability layer (i.e. mapping). In a network all the peers manages its own conventional databases and executes a queer and updates the related data in other peers also. It also joins the databases from one or more peer and form its own materialized view. In this paper we shoe the client-server system in which any of the peers can act as a client requesting a data from the other peer acting as server. We mainly focus on how to maintain a consistent execution view of concurrent transactions in peers without a global transaction coordinator. Since there is no global transaction coordinator and each peer act independently by following the client-server system mechanism. We also mention the problems arise during maintaining the concurrent execution and to solve them we have given two methods first is that only one peer can carry out the concurrent execution and the second one is the OTM optimistic ticket method.
Keywords- Peer to Peer, Data Sharing, Mapping, Transactions

Citation: Mishra A.K., et al. (2012) Processing Of Concurrent Transactions in a Peer To Peer Database Network System. International Journal of Networking, ISSN: 2249-278X & E-ISSN: 2249-2798, Volume 2, Issue 1, pp.-40-43.

Introduction

A peer to peer database network is a collection of peers in which each participates in sharing its database to the other peer. The local database at the each peer is called the peer database and the database which it request from the other peer is called referenced database. Each peer manages its own database independently and maintains its own schema of the relational database. Database in each peer belongs to the database in other peer in some while mean so that for mapping in pair [1].In order to access the data from other resources i.e. peers a globally integrated schema is prepared so that for the sake of convenience that which type of data is stored on which peer. A peer-to-peer (P2P) distributed system is one in which participants rely on one another for service, rather than solely relying on dedicated and often centralized infrastructure. Instead of strictly decomposing the system into clients (which consume services) and servers (which provide them), peers in the system can elect to provide services as well as consume them. The membership of a P2P system is relatively unpredictable: service is provided by the peers that happen to be participating at any given time [1] [5]

For creating a global database, each local source defines an export schema, which describes the data it is willing to share with others [8]. The global database is the union of all the export schemas. Authors in Ref. [6] proposed five-layer schema architecture for loosely-coupled federated database systems. In the architecture there is no federated schema or central administrator. The owners of the independent database systems are responsible for creating and maintaining their own federated schema(s) [2]. In this architecture, the lowest layer (first layer) consists of the local source schemas. The second layer consists of component schemas, which are translations of the source schemas into a common data model. The third layer is comprised of export schemas. Through the export schema, a source describes which part of its data it is willing to share with others. The federated schema integrates multiple export schemas and its information is, in turn, filtered through the external schemas. Each application accesses

the global database by the definition of an external schema. It is also assumed that in a FDBS sources are to be stable and unchanging throughout the system's lifetime. However, in a P2PDBN, sources are loosely-coupled and data vocabularies of peers may be different but may represent the same real-world entities. Making a virtual global schema from the schemas of peers is not possible due to the pair-wise mappings between peers and the dynamic behavior of peers.
Traditionally, data integration and exchange between heterogeneous data sources is provided mainly through the use of views, i.e., queries that map and restructure data between the heterogeneous schemas [13, 20]. Since queries depend on the underlying schemas, to correctly restructure and map data, the sources must be willing to share their schemas and cooperate in establishing and managing the queries. In our work, we consider peer-to-peer settings in which such close cooperation is either not desirable (perhaps for privacy reasons) or not feasible (perhaps due to resource limitations or the dynamic nature of the data structures) [11, 17].

General Execution scenario of transaction
The figure shows the peers are connected in a P2P network and each peer has a database close to it along with a set of mapping tables. The figure also shows that a peer can set off both local and global transactions; for example, peer 1 initiates a local transaction L1 as well as a global transaction T1. The global transaction is propagated throughout the network k to peer 2 after being translated as T1 2.

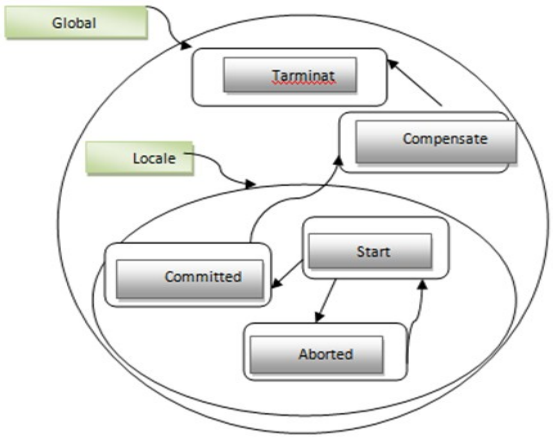

Fig 1- Transactions Process Database Network in Peer To Peer System.

In transactions process database network in a peer-to-peer system fig**.** (1). States of a transaction a scattered, transactions are executed under the control of the GTM. In contrast, a P2P transaction model is built on a network of peers without a global transaction controller..
The transaction process database network in peer to peer system. Each local database management organization conserve the atomicity, consistency, isolation and durability (ACID) properties of transactions and ensures serializability of the local agenda using the local concurrency protocol since the LDBSs are pre-existing. There is also a difference between a scattered transaction model and P2P transaction model. In a distributed transaction model global level transactions are issue to the global transaction administrator (GTM) and are decaying into a set of sub transactions to be separately submitted to the corresponding LDBSs. In P2P transaction model, a global transaction is not decaying but rather is propagated as an entire transaction. A peer, after complete transaction locally, ahead the entire transaction to its acquaintances. The isolated peer that receives the transaction & submitted by local users[7].

Types of Transaction
Local
Local states symbolize the different sates of a transaction during its local execution in a peer. There are three different local states, namely, *start*, aborted *(A)*,and committed *(C)*. The start state symbolizes the beginning of execution of a transaction in a peer. A transaction can be locally-aborted or locally-committed in a peer. If a transaction is successfully executed in a peer, it is committed by the local transaction manager of the peer and the state of the transaction changes from start to locally-committed state. A change of state is denoted by an arrow in the Figure 4.1.1. However, if the transaction is aborted due to the failure of execution, the state becomes locally-aborted.

Global states
The global states symbolize the execution status of a transaction in a peer to peer network. There are two states in this group, namely, *terminate* and *compensate*. The terminate state of a transaction symbolizes that the transaction is successfully committed by the participating peers in the network. If a transaction is terminated, all the information related to the execution of the transaction in the network is deleted from the participating peers. The compensate transaction semantically undoes the effect of the execution of the transaction.

Algorithm for mapping

Consider a path θ = P1 P2..... Pn of peers and let U_i be the set of attributes in peer P_i, $1 \leq i \leq n$. Let $\in$ denote the set of mapping constraints over path θ. Two more notions are necessary for our purposes. The first notion is that of an extension of a mapping constraint. Specifically, given a mapping constraint μ : X $\xrightarrow{m}$ Y , we define the extension of μ , denoted as *ext(μ)*, to be:

ext(μ)={ $\rho(t) | t \in m$ *and* ρ *is evaluation over m }*

Furthermore, we say that μ is a cover of a set of mapping constraints $\in$ over U if

1. $\in$ is consistent if and only if there exists t 2 *ext(μ)*

2. For every mapping constraint μ' : X $\xrightarrow{m'}$ Y , $\in=$ μ' if and only if

$ext(\mu) \subseteq ext(\mu')$.
The algorithm presented in the following paragraphs accepts as input a path Θ, a set $\in$ of mapping constraints over path Θ and two sets of attributes X $\subseteq$ U1, Y $\subseteq$ Un in peers P1 and Pn, respectively. Then, it computes a mapping constraint $\mu : X \xrightarrow{m'} Y$ that is a cover of the set $\in$ of constraints. As such, the algorithm can be used to solve both the inference and the consistency problems. For the inference problem, given $\in$ and a mapping $\mu' : X \xrightarrow{m'} Y$. To solve this problem, it is suficient to run the proposed algorithm and check whether $ext(\mu) \subseteq ext(\mu')$. The check, due to Condition 2 above, provides an answer to the inference problem. For the consistency problem, we run our algorithm as before with the exception of sets X and Y which, in this case, are all the attributes in peers P1 and Pn respectively, i.e., X = U1 and Y = Un. At the end of the algorithm, we can check whether Condition 1 above is satisfied. If this is the case, set $\in$ is consistent [3].

Related work

There is an rising significance in the creation of peer data management systems [4], [9], [10] which includes establishing and maintaining mappings between peers an processing of queries using appropriate propagation techniques. However, the systems do not consider the case of obtaining consistent answers to queries in the presence of the situations where peers may be inconsistent wrt mappings. The inconsistent situations are very common in peer data management systems since peers are autonomous and store data independently. Moreover, mappings may be changed in time. Therefore, it is necessary to find an approach to obtain consistent answers of queries in the systems without changing the physical data in peers to solve inconsistencies. Authors in [14], [15] introduced a semantics for obtaining consistent answers in peer data exchange systems. The semantics utilize the concepts of repair [5] semantics that is proposed to obtain consistent answers in inconsistent databases. Our work also goes in this direction. However, we consider a system where peers are related with value-level constraints that are created by mapping tables. Authors in proposed a query translation algorithm considering that the peers are related with value-level constraints. However, the authors do not consider the case of consistent query answering.

Experimental Results

Implementation for peer to peer processing for making a pair wise mapping in database network

Below gives the pair wise mapping with network for that a simple java code is given which firstly specifies a person working at one peer to be authenticated himself first so that no unauthorized access of data could be done.

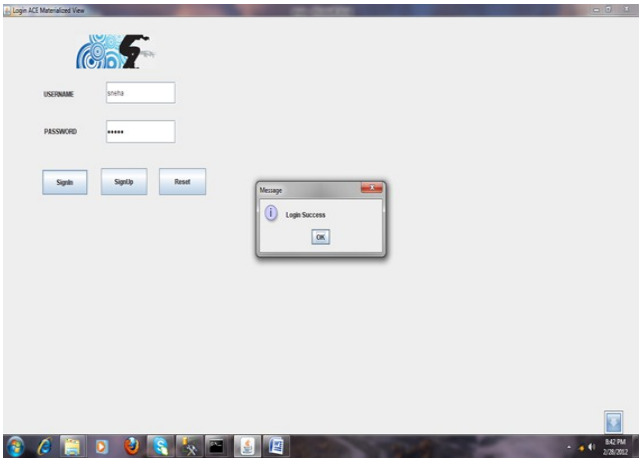

Fig. 2- Authentication for user

Connecting with server

For the local database having no data:
Total records selected:0
Total time required:0.0ms
Executing Query without using Materialized View
Query Executed Successfully
Total records selected:0
Total time required:0.0ms

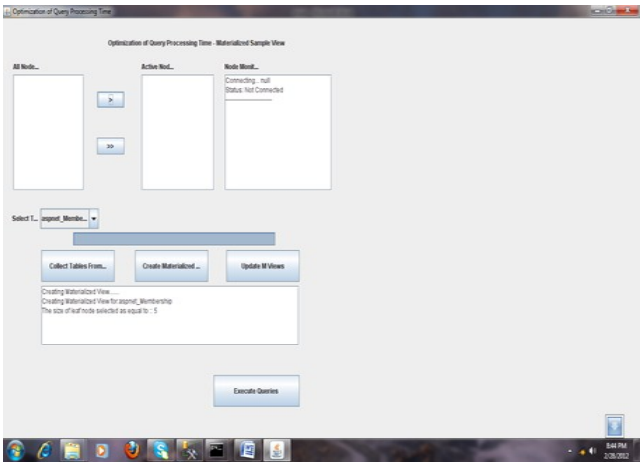

Fig. 3- connection with server as active nodes

creating materialised view of
Creating Materialized View
Creating Materialized View for:USERDETAILS
The size of leaf node selected as equal to :: 5
Materialized View created:USERDETAILSView

After connecting with sever it checks the active nodes i.e. peer in database network the single mapping is done by selecting the first add button in above Fig(3). After adding all the active nodes status of the each node is observed in the next textfield if one of the peer does not give the permission to connect with its database then it shows the status not connected as shown in the above fig 3. Here we consider the connection of local database then selecting one of its relational schema tables are collected by selecting collect tables from then the information is shown in the below textfield

which tupples are used then creating its materialized view the user can save it as its own materialized view and then updating data in that materialized view will update the data in other peers also.

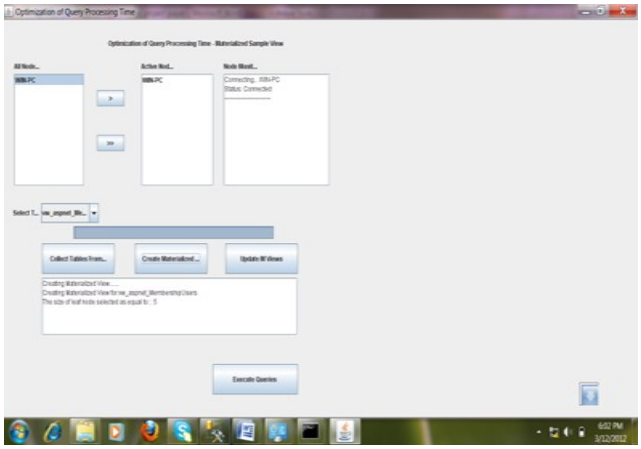

Fig. 4- Mapping and executing queries in network

Total records selected:14
Total time required:4.0ms
Executing Query without using Materialized View
Query Executed Successfully
Total records selected:14
Total time required:4.0ms

After connecting to the one of the peer by mapping algorithm shown above we can execute our own queries in above fig 4 it is shown so by selecting the query type select we can execute our own queries and make changes in any of the database of the connected peer one QPDB database file is created where query details and user details are shown which user is currently working and which changes are made by him in the database.

Conclusion

In this paper, we introduce a transaction execution model for a peer to peer database network where sources are heterogeneous and instance-level mappings are used to associate data from different sources. Our approach is scalable because a peer doesn't need any global knowledge of the system and there is no global coordinator. Transactions are processed by each peer independently and consistency is maintained recursively through acquaintances. A peer only ensures the serializability of its immediate acquaintances by ensuring acquaintance-level serializability. Mainly, we contribute the following:

- We analyze the execution of transactions in a peer to peer database network.
- We introduce a mapping algorithm to ensure the consistency of a peer to peer database network during the concurrent execution of transactions initiated from a single peer.

We propose two approaches ensuring a global consistent execution of transactions without violating the autonomy of LDBSs. A future goal is to investigate the transaction processing when global transactions are initiated from many peers that need to be executed concurrently in the system and to analyze the correctness criteria for such executions.

References

[1] Mehedi Masud, Iluju Kiringa, *International journal of data knowledge and engineering*.

[2] Zaihrayeu I. (2006) *Towards Peer-to-Peer Information Management Systems. University of Trento (Italy), PhD thesis*.

[3] Kementsietsidis A., Arenas M. and Miller R.J. *Mapping Data in Peer to Peer Systems: Semantics and Algorithmic Issues.*

[4] Arenas M., Kantere V., Kementsietsidis A., Kiringa I., Miller R.J. and Mylopoulos J. (2003) *The Hyperion Project: From Data Integration to Data Coordination. In SIGMOD RECORD*.

[5] Arenas M., Bertossi L. and Chomicki J. (1999) *PODC.*

[6] Sheth A.P., Larson J.A. (1990) *ACM Computing Surveys,* 22 (3), 183-236.

[7] Mehedi Masud and Sultan Aljahdali, *International Journal of Intelligent Information and Database Systems*, x(x), xxxx1.

[8] Ozsu M.T., Valduriez P. (1999) *Principles of Distributed Database Systems, 2nd edition*.

[9] Tatarinov I., Ives Z., Madhavan J., Halevy A., Suciu D., Dalvi N. and Dong X. (2003) *ICDE*.

[10] Ng W.S., Ooi B.C., Tan K.L. and Zhou A.Y. (2003) *Data Engineering*.

[11] Bernstein P., Giunchiglia F., Kementsietsidis A., Mylopoulos J., Sera_ni L. and Zaihrayeu I. (2002) *WebDB*.

[12] Georgakopoulos D., Rusinkiewicz M., Sheth A. (1994) *IEEE Transactions on Knowledge and Data Engineering,* 6(1), 166-180.

[13] Chang C.C.K. and Garcia-Molina H. (1999) *SIGMOD*, 335, 346.

[14] Bertossi L. and Bravo L. (2007) *International Conference on Logic for Programming, Artificial Intelligence and Reasoning*.

[15] Bertossi L. and Bravo L. (2004) *International Workshop on Peer-to- Peer Computing and DataBases*.

[16] Silberschatz, Korsth, Sudarshan, *Database System Concepts, 4th edition,* 565-590.

[17] Gribble S., Halevy A., Ives Z., Rodrig M. and Suciu D. (2001) *WebDB*.

[18] Sheth A.P., Larson J.A. (1990) *ACM Computing Surveys,* 22 (3), 183-236.

[19] Ozsu M.T., Valduriez P. (1999) *Principles of Distributed Database Systems", 2nd edition.*

[20] Lenzerini M. (2002) *PODS*, 233-246.