Performance optimization algorithm of a distributed database with a hierarchical network topology

Arij Al Adel
Post-graduate of Faculty of Radio Engineering and Cybernetics
Moscow Institute of Physics and Technology(national research institute)
Moscow, Russia
areejmayaraladel@yahoo.com

Aleksandr V. Belov
Department of the Applied Mathematics
National Research University "Higher School of Economics"
Moscow, Russia
avbelov@hse.ru

Abstract: This paper addresses analyzing the data flows that appear when distributed database works. An algorithm for optimizing database replication is proposed. As a protocol data replication two-phase commit protocol (2PC) with two levels of lock records (shared lock (Shared), and an exclusive lock (Exclusive)) is considered.

Keywords: enterprise information system, distributed database, two-phase commit protocol (2PC), optimization algorithm

1 Introduction

Modern enterprise information systems (EIS) applied for enterprises automation are usually designed using databases. Enterprises with a distributed structure naturally faced with the automation of the entire enterprise challenge. Usually, the automation of separate structural subdivisions through the simulation of business processes can improve the situation within the subdivisions but at the same time there is the problem of data synchronization of various systems. The solution to this problem is to develop a distributed EIS. The EIS includes a distributed database [1] consisting of a set of local databases, server equipment, system software, client computers and application software. The functioning EIS may limit or even determine the speed at which business processes are performed. Therefore, the task of optimizing the performance of EIS becomes very important. The EIS database at any given time contains all the information about automated business processes. The business process should be implemented in the most efficient manner. Thus the performance of EIS significantly affects the efficiency of enterprise business processes. There are many methods for evaluating the performance of EIS [2, 3, 4]. The main criterion for the effectiveness of EIS used in these methods is economic efficiency. Due to the fact that for many business processes related to customer service, production operation, etc., the main indicator of their effectiveness is T_{EIS} - the time spent by EIS on the operation.

$$T_{EIS} = T_{CL} + T_{TL} + T_{SL},$$

Where,
$T_{CL}$is the time taken by the application executed on the client computer to complete the business process operation (Client level),
T_{TL} - the time taken to transfer data between the client computer and the database server (Transportation level),
T_{SL} - the time taken by the server to complete requests / transactions (Server level).
For geographically distributed enterprises business processes are also distributed. Therefore, to increase the efficiency of geographically distributed EISs it is necessary to optimize the T_{SL} time by constructing a replication scheme.
Now there are several methods which aim to optimize database performance. These methods can be applied at all stages of system design: at the knowledge domain analysis, at the design of the logical structure of the database, at the synthesis of the physical database model. Knowledge domain analysis affects the logical structure of the database, the database logical

structure has a direct impact on the algorithms of software applications and the functioning of the system software [5]. Therefore, the optimization process is carried out consistently. At each subsequent stage the results of the previous one are used. Despite the undoubted theoretical and practical interest, a common disadvantage of all methods is that to solve the optimization problem, information is needed about the work of all components of the EIS architecture the collection of which is difficult.

The proposed algorithm has several advantages: it does not have restrictions on the number and structure of databases, it requires a minimum amount of input data, it is easy to implement. It can serve as a basis for designing the network topology optimization algorithms it makes it possible to evaluate the design solution in the early stages of development.

2 Problem Statement

Consider the hierarchical structure of the enterprise network. In general database servers located at network nodes may be interconnected by a multiple channels forming a graph. A tree network is very important case and it is often used in practice. This type of network topology is used for vertically integrated companies. For example, the network of the gas stations have a 3-tier administration structure. Suppose one server as root server. It is connected to data transfer channels with subordinate servers. Child servers are not interconnected by information transfer channels. Child servers, in turn, may have other child servers. The root server always contains a database, which we will call the Central Database (CDB). The remaining databases will be called Remote Database (RDB). Figure 1 shows hierarchical topology of a distributed database.

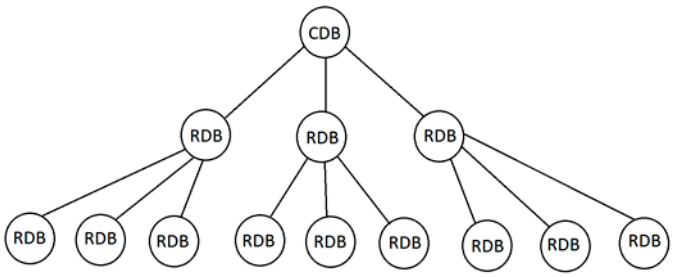

Figure 1 - Hierarchical topology of a distributed database

CDB is to store and process all information about the enterprise activities. Its structure is the same for the remote databases. Remote databases contain a subset of information from the CDB. Any of the remote database can be retrieved from the information contained in the CDB. All servers run a special program - the database server. Each of these programs can access the server's RAM, server disk memory, receive and transmit network requests. The collaboration of these programs allows to perform elementary operations on the database [6, 7].

We provide a formal description of the distributed database model.

Let the Central database D consists of some set of tables:

$D = \left\{ d_i / i = \overline{1,I} \right\}$

All records of one table have the same number of attributes. Let's call the number of attributes in the table - “Table length”.

$K = \left\{ k_i / i = \overline{1,I} \right\}$ — set of the Table length

$N = \left\{ n_i / i = \overline{1,I} \right\}$ — number of records in table i

$Q = \left\{ q_r / r = \overline{0,R} \right\}$ — set of servers (network nodes)

Let q_0 the root server.

$H = [h_{r_1 r_2}]$ — the correspondent matrix to the topology of the computer network where $h_{r_1 r_2}$ takes the following values:

$$\begin{cases} 1, & \text{when } r_1 <> r_2 \text{ and } r_1 \text{ is the immediate parent of the server } r_2, \text{ or when } r_1 = r_2 \\ 0, & \text{when } r_1 <> r_2 \text{ and } r_1 \text{ is not the immediate parent of the server } r_2 \end{cases}$$

Depending on the replication protocol used various elementary operations on database records can be distinguished.

Let $W = \left\{ w_s / s = \overline{1,S} \right\}$ — a set of elementary operations on database records.

Consider the operation of a distributed database over the period T. Let every client be connected to any of the servers. Each client performs different tasks. The application sends requests to the server directly connected to the client’s computer.

Let $\Lambda = \left\| \xi_{irs} \right\|$ — number of elementary operations w_s over a period T to be performed on a table d_i on the node q_r.

We introduce a vector of variables that describes the deployment of tables on every server.

$X = [x_{ir}]$, where $x_{ir} = 1$ if table d_i belongs to the node q_r, $x_{ir} = 0$ otherwise. Since all tables are stored on the root server, so we have $x_{i0} = 1$.

We introduce the following notation:

$B = \|b_{ir}\|$ — the total amount of transmitted data over the channel, that bind the node q_r with the parent node when performing operations on the table d_i over a period T.

$C = \|c_{ir}\|$ — total amount of data received by channel, that bind node q_r with the parent node when performing operations on the table d_i over the period T.

The task of optimizing the performance of distributed DB is formulated as follows: it is required to find such an arrangement of tables $X = [x_{ir}]$ at which the minimum of the transmitted data is reached

$$\min \sum_{\forall i,r} (b_{ir} + c_{ir}) \qquad (1)$$

3 Designing a performance optimization algorithm for a distributed database

To get the values $B = \|b_{ir}\|$ and $C = \|c_{ir}\|$ we analyze the data flows resulted in the execution of one elementary operation on the database. Each elementary operation leads to the data transfer over each channel of the network. Let $Send\ (d_i, r, q_{r_j}, w_s, X)$ - the amount of transmitted data, and $Rcv\ (d_i, r, q_{r_j}, w_s, X)$ the amount of received data, respectively, on the channel binding the server q_r, located on the child node, with the server on the parent node, when performing an elementary operation w_s on table d_i, which was initiated from the server q_{r_j} with a given replication table X, then

$$b_{ir} = \sum_{\forall r_j, s} \xi_{ir_j s} \cdot Send(d_i, r, q_{r_j}, w_s, X),\ c_{ir} = \sum_{\forall r_j, s} \xi_{ir_j s} \cdot Rcv(d_i, r, q_{r_j}, w_s, X) \qquad (2)$$

Consider as a protocol data replication two-phase commit protocol (2PC) with shared and exclusive locks. [8].

Let a transaction be initiated on one of the nodes, which changes the state of the data located in the database on the network nodes Participants in a transaction are interacting processes that run on servers which in turn contain database table records. One of the processes is selected and referred to as "Coordinator". Coordinator sends a request to the involved processes. All processes respond "committed" or "failed", depending on whether the requested operation can be performed. When responses from all processes came the Coordinator takes one of the decisions “commit a transaction” or “rollback a transaction” and sends the decision made to all participants in the transaction. Later when analyzing the data flow it is assumed that all transactions are successfully completed [9]. The distributed transaction is shown in Figure – 2.

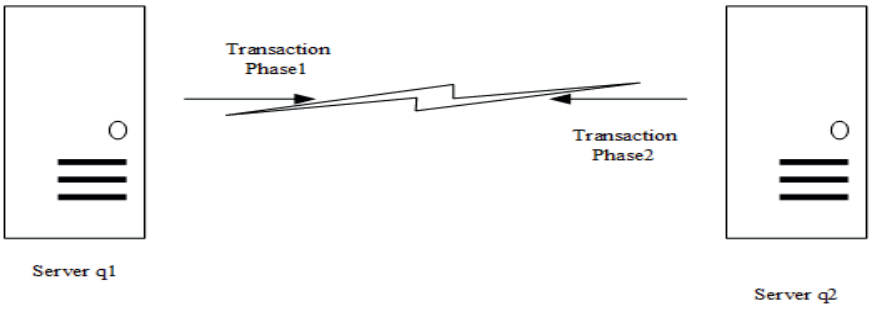

Figure 2 - Performing a distributed transaction diagram

Consider the amount of data transmitted when performing operations on the data. Suppose that a fixed-length service packet L is used to transmit information which will probably be supplemented with data volume k_i about the record on which the operation is performed. Ten elementary operations on a distributed database are introduced and shown in Table 1.

Table 1. Elementary operations on a distributed database

№ operation	Operation 1	Operation 2	Operation 3	Operation 4	Operation 5
Operation name	Record creation	Record reading without blocking	Reading with exclusive lock	Deleting an entry (with the already superimposed exclusive lock)	Replacing the record (with the already superimposed exclusive lock)
№ п/п	Operation 6	Operation 7	Operation 8	Operation 9	Operation 10

Operation name	Removing an exclusive lock	Reading with shared lock	Removing shared lock	Overlaying an excluded record lock with an already superimposed shared lock	Reducing the lock level to a shared one for the record with superimposed exclusive lock

Consider as an example the operation 3 — reading with exclusive lock on table entry d_i. Suppose the network consists of five database servers. Moreover the servers q_0, q_1 q_2, q_4 — contain a table record, q_3 — does not contain a table record ($x_{i0}=1$, $x_{i1}=1$ $x_{i2}=1$, $x_{i3}=0$, $x_{i4}=1$). The transaction process is shown in Figure 3. The corresponding amounts of transmitted data at each stage over the transaction are shown in Table 2.

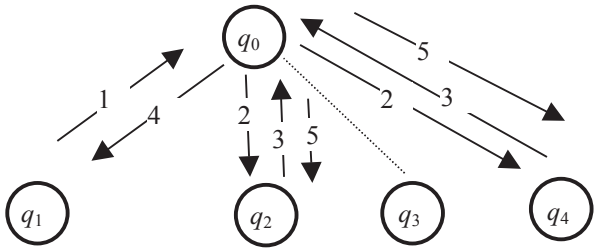

Figure 3 - Transaction flow diagram

Table 2. Amount of data transmitted during the transaction "Reading the exclusive lock"

Transaction Phase	q_1 contains a database record $x_{i1}=1$	q_1 does not contain a database record $x_{i1}=0$
1. Request to the parent server to get a record	L	L
2. Inquiry about availability for child servers	L	L
3. Get answers about the readiness of the child servers	L	L
4. Commit command to the server transaction initiator and, if necessary data transmission	L	$L + k_i$
5. Commit command to other child nodes	L	L

Based on the data in Table 2, it is possible to get the total amount of transmitted data over each channel. All elementary database operations are treated in the same way. The results are shown in Table 3.

Table 3. The amount of transmitted information through the channel binding (which bind) the server q_{r_j} with parent server q_r

The server q_r contains the table d_i	yes $x_{ir}=1$		no $x_{ir}=0$		yes $x_{ir}=1$		no	
Data transfer direction	To the server CDB *Send*	From the server CDB *Rcv*	To the server CDB *Send*	From the server CDB *Rcv*	To the server CDB *Send*	From the server CDB *Rcv*	To the server CDB *Send*	*From the server CDB Rcv*
Column number	1	2	3	4	5	6	7	8
Elementary operation								
Operation 1	$L + k_i$	L	$L + k_i$	L	L	$2\ L + k_i$	0	0
Operation 2	0	0	L	$L + k_i$	0	0	0	0
Operation 3	L	L	L	$L + k_i$	L	$2\ L$	0	0
Operation 4	L	L	L	L	L	$2\ L$	0	0
Operation 5	$L + k_i$	L	$L + k_i$	L	L	$2\ L + k_i$	0	0

Operation 6	L	L	L	L	L	$2\ L$	0	0
Operation 7	0	0	L	$L\ +\ k_i$	0	0	0	0
Operation 8	0	0	L	L	0	0	0	0
Operation 9	L	L	L	L	L	$2\ L$	0	0
Operation 10	L	L	L	L	L	L	0	0

Looking at table 3 it is clear that the amount of transmitted data have the form $\alpha \cdot L + \beta \cdot k_i$ where α and β non-negative integer coefficients. A convenient representation of data amount is matrixes of coefficients $\mathrm{A} = \|\alpha_{st}\|$ and $\mathrm{B} = \|\beta_{st}\|$ where $s = 1,\ldots,10$ — number of elementary operation on database, $t = 1,\ldots,8$ — column number in the Table *2*.

In this way

$Send(d_i, r, q_{r_j}, w_s, X) = \alpha_{st} \cdot L + \beta_{st} \cdot k_i$

$Rcv(d_i, r, q_{r_j}, w_s, X) = \alpha_{st} \cdot L + \beta_{st} \cdot k_i$,

where column number t determined by the value of the function arguments and the name of the function. Let function *notlog* (condition) takes the value 0 if the condition takes the value «true» and value 1 if the condition is "false".

Finally the total amount of transmitted data and received over the channel binding the node q_r with parent node:

$$b_{ir} = \sum_{\forall r_j, s} \xi_{ir_j s} \cdot (\alpha_{st+1} \cdot L + \beta_{st+1} \cdot k_i),\ c_{ir} = \sum_{\forall r_j, s} \xi_{ir_j s} \cdot (\alpha_{st+2} \cdot L + \beta_{st+2} \cdot k_i) \quad (3)$$

where $t = notlog$ (q_{r_j} is subordinate to the node q_r)

essentially b_{ir} and c_{ir} depend only on the value of x_{ir} and do not depend on the location of the tables on other servers.

The following optimization algorithm is proposed.

Input data for algorithm

$K = \left\{k_i / i = \overline{1, I}\right\}$ — table length

L — service packet length

$F = [f_{r_1 r_2}]$ — computer network topology

$\Lambda = \|\xi_{irs}\|$ — number of elementary operations

$\mathrm{A} = \|\alpha_{st}\|$ и $\mathrm{B} = \|\beta_{st}\|$ — coefficient matrixes

Result:

$X = \|x_{ir}\|$ — table replication

The algorithm is as follows:

Step 1. The initial values of the replication scheme are set.

$X = \|x_{ir}\|$, $x_{i0} = 1$ and $x_{ir} = 0\ \ \forall\ r \neq 0$

$D_{lookup} = \{1,\ldots,I\}$

Step 2. A database table that have not yet been reviewed is selected. $i_1 \in D_{lookup}$, $D_{lookup} = D_{lookup} \setminus \{i_1\}$. If the set $D_{lookup} = \varnothing$ empty the algorithm terminates go to step 8.

Step 3. Servers set that need to be considered is being initialized .

$R_{lookup} = \{0\}$

Step 4. B current server is selected $r_1 \in R_{lookup}$, $R_{lookup} = R_{lookup} \setminus \{r_1\}$

If the set $R_{lookup} = \varnothing$ empty go to step 2.

Step 5. One of the information channels that bind the current server is selected r_1 containing database table i_1 with directly subordinate server r_2 . If all channels are reviewed move to step 4.

Step 6. Data set is processed $b_{i_1 r_2} + c_{i_1 r_2}$, transmitted over the channel $r_1 r_2$ in cases where the table i_1 will not replicate to the server $x_{i_1 r_2} = 0$ and when the table will be replicated to the child server $x_{i_1 r_2} = 1$

Step 7. Based on the calculated quantities a decision is made to replicate the table. i_1 to server r_2 . The table is replicated to the child server r_2 if this can lead to a decrease in the amount of data transmitted over the channel $r_1 r_2$.

If $x_{i_1 r_2} = 0$ then go to step 5.

If $x_{i_1 r_2} = 1$ first we add the server r_2 к списку серверов, которые необходимо рассмотреть $R_{lookup} = R_{lookup} \cup \{r_2\}$ go to step 5.

Step 8. Completion of the algorithm work.

Maintain optimal replication scheme values $X = \|x_{ir}\|$. with this, minimum total amount of transmitted data is reached $min \sum_{\forall i,r}(b_{ir} + c_{ir})$.

4 Experiments

To test the operability of the proposed approach we evaluated the performance of the replication scheme constructed in accordance with the algorithm considered above.

The computational experiment was conducted on the basis of EIS Apparel Business Management Systems. To register elementary operations on the EIS database a tool was developed in C ++ with the help of which the total quantities of transmitted and received data and the time for their transmission were measured.

As a business transaction the operation "Create Purchase Order" was used to purchase raw material. The information model of the Purchase Order is two related tables one of which contains general information about the order (Header_PO), and the other contains detailed information about the purchased material: material name, quantity, unit price etc. (Line_PO).

Since the operation for the purchase of material involves two nodes: Head Office (HO), Warehouse (WH) in which transactions for creating PO are carried out the parameters used in the algorithm were determined. Two EIS construction schemes were considered:

Scheme 1 - On the server in HO - there is a database and the WH server does not contain a database

Scheme 2 - On both objects; HO and WH are used database servers.

The calculation formulas are given in table 4.

Table 4. Calculation formulas for determine data transfer time

Scheme Type	**Server**	**Data transfer time T_{SL}, sec**
Scheme 1	Exclude DB	$\frac{907N_L^2 + 14065N_L + 22544}{V_{SL}}$
Scheme 2	Include DB	$\frac{1157N_L + 2083}{V_{SL}}$

N_L - Number of PO lines created in PO, V_{SL} - channel bandwidth, numerical coefficients were determined by recording the amount of transmitted and received data

The results of numerical modeling for specific parameter values are given in table 5.

Table 5. Results of the computer simulation

Number of PO lines created in PO, N_L	**Data transfer time T_{SL}, sec**	
	Scheme1	**Scheme2**
1	0,29	0,02
2	0,41	0,03
3	0,56	0,04
4	0,71	0,05
5	0,88	0,06
6	1,06	0,07
7	1,26	0,08

The results of numerical experiments showed the efficiency of the replication scheme between database servers when implementing business processes in the distributed EIS.

5 Conclusion

The direction of further research may be the consideration of unreliable information transfer channels, equipment failures, determination of the time of user requests.The proposed approach can be used in the design of distributed registries using blockchain technology.

References

1. M. T. Özsu and P. Valduriez, Principles of Distributed Databases (3rd edition) (2011), Springer, ISBN 978-1-4419-8833-1
2. Nicolaou A. I., Firm performance effects in relation to the implementation and use of enterprise resource planning systems. J Information System, 18, 79-105, 2014
3. Hawary A., Heeks R. Explaining ERP failure in a developing country: a Jordanian case study. J Enterprise Information Manage, 23 (2), 135-160, 2010
4. Guedria W., Naudet Y., Chen D. maturity model for enterprise interoperability. Enterprise Information System, 9 (1), 1-28, 2015

5. Antonios Makris, Konstantinos Tserpes, and Dimosthenis Anagnostopoulos. A novel object placement protocol for minimizing the average response time of get operations in distributed key-value stores. In Big Data (Big Data) 2017, IEEE International Conference on, pages 3196–3205. IEEE, 2017.
6. Bermbach, D., Kuhlenkamp, J.: Consistency in distributed storage systems. Networked Systems, pp. 175-189. Springer, 2013
7. Domaschka J., Hauser C.B., Erb, B.: Reliability and availability properties of distributed database systems. In: Enterprise Distributed Object Computing Conference (EDOC 2014), IEEE 18th International. pp. 226-233. IEEE, 2014
8. https://docs.oracle.com/cd/B19306_01/server.102/b14231/ds_txns.htmhttps://docs.oracle.com/cd/B19306_01/server.102/b14231/ds_txns.htm
9. B. M. M. Alom, F. Henskens, and M. Hannaford, "Deadlock Detection Views of Distributed Database," in International conference on Information Technology & New Generartion (ITNG- 2009) Las Vegas, USA: IEEE Computer Society, 2009