Optimization for a Distributed Query

Parminder Kaur

School of Computer Science & Engineering, Punjab, India

Abstract: ***Distributed databases are the outcomes of top notch technology advances, high speed computer networks further facilitated its growth and its suitability in satisfying various businesses needs make it more popular. The major area of proposed work is query optimization in distributed database systems. The objective of query optimization is to execute the query with minimum cost. In order to perform join operation, two sub queries involving data from multiple sites has to be transmitted from one site to other. As data resides at different sites in a distributed database environment, so to acquire a specific type of data; subdivision of a query into its sub-parts (sub-queries) is required and those sub-queries needs to be executed at different data sites. In some cases combination of data from two or more different sites may be required. To attain this goal a join operator is used. But using join is not always advantageous in terms of cost as it may sometimes result in more communication cost in cases when complete relation is not desired for join operation. In such scenario communication cost involved between two sites can be reduced using other forms of joins like inner join. Inner join is also not always useful. So a need of finding the appropriate strategy to decide and assign join operations arises. In this paper join operator allocation has been done dynamically by dynamically calculating percentage participations for joins and inner joins.***

Keywords: Query, distributed database, network, cost, manipulation**.**

1. Introduction

Data is the base of whole world of growing organizations in today's world and managing data is one of the most trivial task. Database Management System are used to manage whole data in organizations. In today's world of universal dependence on information systems, every user of the system whether an employee or a customer need access to company's databases. Database is managed using two approaches known as Centralized Database Management System and Distributed Database Management System [8]. Distributed database is a collection of logically interrelated databases that can be stored at different computer network sites. The objective of a distributed database management system (DDBMS) [8] is to control the management of a distributed database (DDB) in such a way that it appears to the user as a centralized database. A distributed Query can reside on network servers on the internet, on the corporate intranets and extranets or on other company networks.

1.1 Database Management System (DBMS):

Database management system (DBMS) [8] is software [9] systems, which are basically collection of interrelated data and allow definition, creation, updating of databases [15]. DBMS are applications that are designed in such a way that they can interact with users, other applications, and database itself to analyze and capture data. DBMS provide a lot of facilities, some of which are:

1.1.1 Data Definition Language [16]

DBMS provide its users facility to define database, using data definition language (DDL). Users can specify structure of database, data types and constraints on data by using DDL.

1.1.2 Data Manipulation Language [17]

DBMS provide its users facility to insert, retrieve, update and delete data by using Data Manipulation Language (DML). DML provide general facility to enquire about data, which is known as query language.

1.1.3 View Mechanism [6]

DDL is also use to define a view. A view is basically a subset of database, but it doesn't form part of physical schema. Each user can have his or her view of database.

1.2 Design of Distributed Database

To plan [7] a distributed database system is one of the most crucial aspect behind the success or failure of such a system. Designing a distributed system involves taking decisions related to the placement of data and programs in system (includes network nodes and network design itself). While designing a distributed system, the main focus is given to the division and placement of data i.e. to the placement of data. The issue that arises while designing a distributed database system are:

a. Why fragment at all
b. How to fragment
c. How much to fragment
d. How to test correctness
e. How to allocate

Two basic strategies used in designing distributed database system are:

- Top-down Approach

Involves designing the system from scratch Used for homogeneous systems.

- Bottom-up Approach

Used for systems where database already exists at some sites The aim is to connect the databases to solve common tasks.

1.3 Query Optimization [4]

A query can be expressed by using two or more equivalent query plan. There can be a huge difference between costs of two alternative plans, depending upon the processing costs at different sites, communication costs etc. Query optimization [4] is the function of determining the most efficient query plan among all, which is performed by query optimizer.

2. Related Work

In this section, some of the important techniques related to the research problem are discussed. The review of literature is a very important part as it links up the studies that have already conducted in the same field. It also puts light on the various aspects which have already been accomplished by researchers and gives us a chance to appreciate the evidences that have already been collected by researchers in their studies and supports the researchers in projecting the current research work in proper perspectives. Along with that, the researchers get chance to learn from the experience of the other studies in the same field and can enrich the proposed study. The research methodologies used by different researchers help. A comprehensive review of a few research papers is given below.

In distributed database systems there are three processes by which data is distributed among various sites, these are: fragmentation, allocation, and replication. Fragmentation process requires empirical knowledge of data access and query frequencies. But Shahidul Islam Khan and Dr. A. S. M. LatifulHoque [12] had proposed a horizontal fragmentation technique that is capable of taking proper fragmentation decision at the initial stage by using the knowledge gathered during requirement analysis phase without the help of empirical data about query execution. It allocates the fragments properly among the sites of DDBMS.

In query processing in distributed systems the main problem is determining the sequence and the sites for performing the set of operations, if the query is subdivided into sub queries that require operations at geographically distributed databases, such that the operating cost for processing the query is minimized. For that B.M. MonjurulAlom, FransHenskens and Michael Hannaford [10] had proposed a technique to process the query with minimum inter site data transfer. The proposed technique is used to determine which relations are to be partitioned into fragments, and where the fragments are to be sent for processing. The technique generally fragments the relations that exist in the predicates (the WHERE condition) of the query. It chooses more than one relation to remain fragmented which exploits parallelism, while replicating the other relations (excluding the fragmented relations) to the sites of the fragmented relations. Thus the communication costs and local processing costs can be reduced due to the reduced size of the fragmented relations and the response time of queries can be improved.

AreeratTrongratsmeethong*et al.* [13] proposed a new join order algorithm called Exhaustive Greedy (EG) algorithm to optimize intermediate result sizes of join queries. Exhaustive search and greedy algorithm are combined and modified to identify good join orders. Exhaustive search algorithm can guarantee the optimal solution as it produces the entire join trees of a join graph. Greedy algorithm used to reduce search space by generating only one query tree in a polynomial time. In order to determine join order selection at subsequent steps, this EG algorithm also updates join graphs to reflect new size of join nodes and new join selectivities of edges associated with the join nodes at each step. In addition, most intermediate result sizes of join queries estimated by the EG algorithm are comparable to the results estimated by the Exhaustive Search algorithm (ESU)that is modified to update join graphs, it is named as ESU Algorithm. Exhaustive algorithm is performed at the beginning to find all the edges in the original join graph then each edge is set as a starting route of the each candidate route. The remaining join orders of each candidate route then are performed by greedy algorithm.

Huang et al. [7] proposed two heuristic algorithms in their research paper to find a near optimal allocation to minimize the total communication costs. He proposed solution for determining the replicated number of each fragments and then of finding a near optimal allocation for all those replicated fragments. Basically, two heuristic algorithms were proposed by them. The first algorithm named Algorithm-1, had three steps. First of all, fragment allocation table is initialized by considering only retrieval requests. Retrieval requests are processed without any communication cost by transferring the target fragment to issuing site. In the second step, the only thing to be considered is update request. Here, the stress is on how to remove fragment copies from the initialized allocation table so that communication cost incurred in update request get reduced. The fragment copy is removed only when the update request cost was much larger than the retrieval request cost. In the last step, it is checked that if there is any fragment which has not been allocated to any site and has been updated by some sites. Then the candidate sites for this fragment are searched form update and frequency matrix and the fragment is allocated to site with least communication cost. In the second algorithm named Algorithm-2, the first and last step were same as Algorithm-1. In the second step, the fragment copy is removed from the initialized allocation plan unless it was not the last copy in the WAN.

Dr.Rajinder Singh Virk, Manik Sharma, Dr.Gurdev Singh [14] In “Analysis of Joins and Semi Joins in a Distributed Database Query” by, the focus is given on computing and analyzing the performance of joins and semi joins in distributed database system. The various metrics that will be considered while analysing performance of join and semi join in distributed database system are Query Cost, Memory used, CPU Cost, Input Output Cost, Sort Operations, Data Transmission, Total Time and Response Time. In short the intention of this study is to analyze the performance and behaviour of join and semi-join approach in distributed database system. From the study it is concluded that the data transmission in a distributed query using semi-join is always lesser than the data transmitted in distributed query using joins operation however data accessed using semi join may be larger than join

operation. No doubt semi-joins implement more operation as compare to join, but it reduces the number of bytes transferred from one site to another to great extent. Further one is able to conclude that semi joins are beneficial if the transmission cost is of main consideration, otherwise joins will be preferred [8].

3. Problem Statement

Query optimization is one of the dominant research subjects in the field of Distributed Database System. The Query optimization can be done by using various techniques like Exhaustive Enumeration, Genetic Algorithm, Ant Colony optimization, etc. There is no dearth of research already done for optimizing query in distributed database system. Majority of researchers have focused on reducing the total cost or response time to make distributed database system an effective one. Communication cost is the cost of shipping the query and its results from the database site to the site where the query originated. This work focuses on minimizing communication cost involved in data transmission Using different kinds of join approachs. There is a need of data transmission from one site to other site. When two sub queries join located at different sites to get the required result Due to this data transmission, communication cost increases. So to reduce this inner joins are used which reduce the relations before transmitting them to other site. But this type of join reduction is not always feasible approach as sometimes all attributes of relation are required for joins operation. In that case it increases the communication cost. In this proposed work query tree will be dynamically generated for different queries. For different queries it will be dynamically decided to choose inner join or join depending upon the dynamically estimated size of relation.

4. Objective of the Study

- The minor objectives under this work are as follows:
- To implement the simulator using MATLAB. The distributed database environment is simulated using Dynamic programming in MATLAB.
- To create a database in MS ACCESS and create a connection string with MATLAB to access the database.
- To study the cost of the query.
- To examine the percentage change in cost of the query processing as the use of joins and inner joins affects the cost of Query Processing in Distributed
- The objective of this research work is to minimize the 'Total Cost of Query', which is represented in terms of time units and refers to use of resources such as CPU Cycles, Disk I/O & Communication Channels by a candidate allocation plan.

5. Methodology

Simulate a Distributed Database Environment in MATLAB for query analysis. SQL query is first translated into equivalent relational algebra expression. The relational algebra expression can be further transformed into an operator tree to visualize the sequence of SQL Query operation from bottom to top in tree structure. Each node shall represent the sub operation like selection, projection or join operation of the query. The node representing the join will be dynamically decided whether to use simple join or inner join, depending upon the one that yields minimum cost. The dynamically created tree data structure is converted into a text file by simulator with various input factors I/O Cost, Communication Cost etc) It acts as an input seed to the simulator. Design and code a Stochastic Simulator in MATLAB simulation environment. The abstract plan of methodology is:

1. Simulate a Distributed Database Environment in MATLAB for query analysis that will consist of OLTP transactions set of experiments in SQL or any high level language (Non procedural language).
2. SQL query is first translated into equivalent relational algebra expression. The relational algebra expression can be further transformed into an operator tree to visualize the sequence of SQL Query operation from bottom to top in tree structure. . A tree defines the order in which operations must be applied in order to produce the result of the query. Intermediate fragments are generated.
3. Each node shall represent the sub operation like selection, projection or join operation of the query. Root node may represent the final operation of a query. The node representing the join will be dynamically decided whether to use simple join or inner join, depending upon the one that yields minimum cost
4. The dynamically created tree data structure is converted into a text file by simulator with various input factors (I/O Cost, Communication Cost, Processing Cost, Data Allocation Matrix) etc. It acts as an input seed to the simulator with various other inputs like cardinality of relations, size of intermediate fragments etc.
5. Dynamically calculate the percentage participation for nodes containing inner join, left join, right join operator. Then check for condition i.e. If (Is percentage participation for inner join (PPIJ) < percentage participation for join (PPJ))
6. Calculate fragment size based on PP for join operation nodes and give this dynamically computed fragment size for join operations as input to the simulator built in MATLAB.

Simulated distributed database environment in MATLAB will use Genetic Algorithm to minimize the objective function which is communication cost involved in transferring data from one site to other while performing joins. Calculate the percentage reduction in communication cost for inner joins against joins for one instance of the database. Dynamically calculate the percentage participation for nodes containing join operator both for joins and inner joins for next instance of database.

6. Results

Simulator is run for different instances of database so as to give image of dynamic database with varied cardinalities each time. It has been observed that when joins and inner joins are used in combinations then communication cost greatly reduces. In GA output the minimum communication cost in different cases is as follows:
Case 1: Using Joins
Minimum communication cost = 72.470890
Case 2: Using Inner joins
Minimum communication cost = 82.502469
Case 3: Using combination of Joins and Inner joins
Minimum communication cost = 47.254146
It shows that when joins and inner joins are used in combination then they are useful. Individually they are incurring more communication cost than when used together.

7. Conclusion

In distributed database systems data is physically distributed among geographically different locations. The main problem in query processing in distributed database system is determining the sub operations like selection, projection, join to be allocated to various sites when a query is divided into sub queries. Selection and projection do not involve any communication cost as there is no need to transmit data while executing these operations. But when there is need to join the data between sites data has to be transmitted from one site to other. This decision of determining the sub operation will be done dynamically in the proposed technique so as to minimize communication cost. The various parameters can be used to reduce the Communication cost using join and inner join are fragment size, size of base table etc.

References

[1] Ahmad, Ishfaq, et al. "Evolutionary algorithms for allocating data in distributed database systems." Distributed and Parallel Databases 11.1 (2002): 5-32
[2] Apers, Peter MG. "Data allocation in distributed database systems." ACM Transactions on Database Systems (TODS) 13.3 (1988): 263-304.
[3] Baiao, Fernanda, Marta Mattoso, and GersonZaverucha. "Horizontal fragmentation in object dbms: New issues and performance evaluation."Performance, Computing, and Communications Conference, 2000. IPCCC'00.Conference Proceeding of the IEEE International.IEEE, 2000.
[4] Bamnote, G. R., and S. S. Agrawal. "Introduction to Query Processing and Optimization."International Journal 3.7 (2013).
[5] Bernstein, Philip A., et al. "Query processing in a system for distributed databases (SDD-1)." ACM Transactions on Database Systems (TODS) 6.4 (1981): 602-625.
[6] Bertino, Elisa. "A view mechanism for object-oriented databases." Advances in Database Technology—EDBT'92. Springer Berlin Heidelberg, 1992.
[7] Huang, Yin-Fu, and Jyh-Her Chen. "Fragment allocation in distributed database design." J. Inf. Sci. Eng. 17.3 (2001): 491-506.
[8] STEFAN, Ileana, and Maricel POPA. "Distributed Database Design–Top-Down Design."
[9] Ray, Chhanda. Distributed Database Systems.Pearson Education India, 2009.
[10] Kunii, Hideko S. "Data Manipulation Language." Graph Data Model. Springer Japan, 1990.29-39.
[11] Robbins, Robert J. "Database Fundamentals." Johns Hopkins University, rrobbins@ gdb.org (1994).
[12] Shahidul Islam Khan and Dr. A. S. M. LatifulHoque, (2010) “A New Technique for Database Fragmentation in Distributed Systems”, IJCA Vol. 5.
[13] Areerat et al, “Exhaustive Greedy Algorithm for Optimizing Intermediate Result Sizes of Join Queries”, IEEE, 2009.
[14] Gurdev Singh and RajinderVirk. “Analysis of Joins and Semi Joins in a Distributed Database Queries” International Journal of Computer Applications 49(16):14-18, July 2012.
[15] Robbins, Robert J, (1994) "Database Fundamentals." Johns Hopkins University, rrobbins@ gdb.org.
[16] Wells, Garth, (2001) "Data Definition Language." Code Centric: T-SQL Programming with Stored Procedures and Triggers. Apress, 2001.35-70.
[17] Kunii, Hideko S. "Data Manipulation Language." Graph Data Model. Springer Japan, 1990.29-39. Pocket Telephone, Inc.