

Executing Joins Dynamically in Distributed Database System Query Optimizer

Sofia Gupta
Research scholar
Department of Computer
Science and Engineering Guru
Nanak Dev University,
Amritsar, Punjab

Rajinder Singh
Asst. professor
Department of Computer
Science and Engineering
GuruNanak Dev University,
Amritsar, Punjab

Tirath Singh
Asst. professor
Department of Computer
Science and Engineering
GuruNanak Dev University,
Amritsar, Punjab

ABSTRACT

In order to join two sub queries involving data from multiple sites data has to be transmitted from one site to other. While transmitting the data within a network, the factors involved in distributed databases is communication cost and amount of data transmitted. To minimize these factors, join operation is used. There are two cases considered in which query processing using join and query processing using semi join are described. The amount of data transfer in case of join is more than in case of semi join. Hence sub operations are executed dynamically to improve the communication cost .

General Terms

Distributed database, query processing, query optimization, data transmitted.

Keywords

Distributed database, query optimization, data transfer, join, semijoin etc.

1. INTRODUCTION

In recent years, there is high rate of research, development and implementation of distributed database management systems. . Distributed database connects several databases that are spread physically across computers in multiple locations by data communication network but it is centralized logically. The benefits provided by distributed processing include increase reliability, availability and localization and reduce communication costs. The two dominant approaches used for storing and managing database are centralized database management system and distributed database management system in which data is placed at central location and distributed over several locations respectively. Independent of the database approach used, one of the important issue in the database is the retrieval of data by using multiple table from central repository in centralized database and from number of sites in distributed database[1].

As data environments grow larger, it becomes difficult to store data at a single site [2]. One of the important problems in distributed systems is the efficient processing of queries in relational DDBMS where data transmission among different sites is involved in processing of query. The parameters like high query response time, sites to access these queries affect the performance of distributed queries. Also, database system performance is effective depends on join operator. Join is the primary target of query optimizer because of the high evaluation costs. The tables reside on different nodes of computer network data must be moved between nodes to join

them [3]. The cost of distributed query involves processing cost and transmission cost.

The transmission of data increases the communication cost. So the optimizer must consider efficient order in which tables are joined in such a way that communication overhead has cut down. There is a problem of finding an efficient join order for a query because query Optimizer has to examine number of existing substitutions queries. One tries to optimize the ordering of join directly whereas another replaces join by combinations of semi joins in order to minimize communication cost [4].

2. QUERY PROCESSING AND QUERY OPTIMIZATION

The performance efficiency of DDBMS is critically concerned to Query processing strategies. The retrieval of queries from different sites in DDB is called distributed query processing because the data is geographically distributed into multiple sites so the processing of a distributed query is composed of the following three phases:

- Local processing phase
- Reduction phase
- Final processing phase

The local processing phase needs local processing such as selections and projections. The reduction phase reduces the size of relations using semi joins and joins. The final processing phase sends all resulting relations to the final site which made result of the query [4]. In query processing, database users specify what data is required rather than specifying the procedure to retrieve the required data. Query processing is more complex and difficult in distributed environment rather than centralized environment. Several factors impact the performance of distributed query processing. These factors are selection of appropriate site, order of operation (like select, project and join) and selection of join method (like semi join, natural join, equi-join etc.). Due to the large number of factors involved, there could be multiple execution plans for a single query. Each plan is associated with a cost and the objective of a distributed query optimizer is to find a plan with lowest possible cost (optimal plan). The execution cost is expressed as a sum of I/O, CPU and communication cost. The goal of query processing is to execute queries effectively in order to minimize the response time and total communication cost. Query optimization is one of most limporttant and valuable stages in query execution[5]. Once query is entered and transform into algebraic form,

query optimizer chooses a best query execution strategy for a given query from set of alternatives that minimizes the cost of query evaluation and speed up the execution. This is known as Cost based optimization.

The cost of a distributed execution strategy can be expressed with the sum of total processing cost (local processing cost involved in all sites) and communication cost. Local processing cost is calculated by CPU cycles, disk I/O. Communication cost is requires to transfer data between participating sites and it is of the main consideration in distributed database systems. Communication cost depends on various factors amount of data exchanged, no. of messages transferred, best site choose for query execution and communication network. The set of query plan is formed by finding possible access path and join algorithms.

3. OBJECTIVES OF JOIN IN DISTRIBUTED DATABASES

Join query execution is more complex in a distributed database than in a centralized database. In a distributed database, joining data on different machines requires the transmission of data. This data transmission could be time-consuming if data transmission data is more. Therefore, distributed database systems need to transfer the data as fast as possible in order to improve join query performance. There are two basic join query execution methods used in the distributed database systems

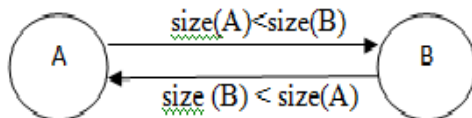


Fig.1 transfer of operands

One method is to transfer the smaller table of two join query participating tables. This method can efficiently perform the join query which the quantity of result is much less than the quantity of two source tables as shown in fig.1. Another way is to transfer two tables in parallel. Parallel transmission can reduce the response time for the join query which the quantity of result is equal to or greater than the quantity of two source tables. The main objective of join query optimization is to reduce the cost of data transmission, small volume of transmitted data and move data in parallel so as to minimize the response time. Therefore, these two methods are not good for all types of join query.

To minimize the amount of data transmission between the sites semi join operator is used. Using Join, entire data most of the tuples in relation participate in the join but in case of semi join the size of relations are reduced [6].

3. RELATED WORK

Sharma et al., 2012 have presented the analysis of the performance of joins and semi joins in distributed database system. The various parameters like CPU cost, IO cost, Query cost, data transmission, total time and response time are considered. The data transmission using semi join is always less than data transmission in distributed query using join operation even so data access using semi join may larger than join operation. Semi join implement more operation than join but it reduces data transferred from one site to another greater

level. Further one is able to conclude that semi join is beneficial if transmission cost is of main consideration otherwise join will be preferred[1].

Pawandeep Kaur et al. presented the join query optimization in Distributed databases. One method for the join query is first to transfer the data from servers to client site and then insert the data into client database, after that join query is performed. Proposed method will directly perform the join query on the client site after fetching from servers site and it will not insert the data into client database. By the proposed method, insertion time of data into client database will be deducted. So, this method will optimize the join query in distributed databases [2].

Sunita M. Mahan et al. 2013 have mentioned the use of semi join operation. Beneficial semi join operation reduces the amount of data transmission required to perform the join sequences. The problem of finding an optimal strategy to minimize data transmission cost in distributed database systems, even with one join attribute is problem determining the optimal sequence of join operations in query optimization leads to exponential complexity. To deal with this problem, there is a need of develop a heuristic approach to solve the problem[7].

Xiaofeng Li et al. have described some common optimization algorithm based on multi relation semi join is submit to apply to this situation that takes buffer zone of distributed database system as the final assembly station of intermediate result query. The experiment proves that query that query optimization algorithm based on multi relation semi join reduces the data volume of intermediate result and effectively decreases the overall cost of network communications.[8]

Lin zhou, Chen et al. have studied the query optimization process based on semi join optimization combined with practical application. A classical SDD-1 algorithm is also introduced which is used for multiple connection and query optimization based on semi join query optimization. There is a comparison of cost and results between two different semi join operation methods. The results show significant impact on query methods can affect the execution speed of system directly. [9]

Kumar et.al have proposed a novel method for query optimization using heuristic based approach to evaluate the efficiency of a query search in the database operations. In the proposed algorithm, a query is searched using the storage file which shows an improvement with respect to the earlier query optimization techniques. Also, the improvement increases once the query goes more complicated and for nesting query. Therefore, heuristic based query optimization is a better approach to query optimization as compared to earlier query optimization techniques[11].

4. EXPERIMENTAL ANALYSIS

In a distributed system, several factors complicate query processing. The first is the cost of transferring data over the network. There is a illustration of this with two simple example queries. Suppose that the EMPLOYEE and DEPARTMENT relations are distributed. According to Figure 1, the size of the EMPLOYEE relation is $100 * 10,000 = 106$ bytes, and the size of the DEPARTMENT relation is $35 * 100 = 3500$ bytes.

4.1 Distributed Query Processing using join

Consider the two example queries there are two relation schemas EMPLOYEE and DEPARTMENT. table EMPLOYEE is at site1 and table2 DEPARTMENT is at site 2. Consider the query “for each employee retrieve the employee name and the name of department for which the employee works”

$Q: \pi_{FNAME, LNAME, DNAME}(EMPLOYEE \bowtie_{DNO=DNUMBER} DEPARTMENT)$

Table1: EMPLOYEE

FNAME	BDATE	ADDRESS	SALARY	DNO.
Sam	16-11-1999	#1234AB	10,000	D1
Joe	12-1-1993	#5643DF	20,000	D1
Tom	22-4-1998	#983FF	15,000	D3
John	4-8-1999	#333AB	30000	D10000

Table2: DEPARTMENT

DNAME	DNUMBER	MGSSN	RGSTRDATE
Electronics	D1	983334447	26-12-2014
Computer Science	D2	983344551	15-3-2007
Architecture	D3	983447877	3-6-2006
Administration	D100	946222234	5-8-2013

The size of the EMP relation is $100 * 10,000 = 106$ bytes and the size of the DEPARTMENT relation is $35 * 100 = 3500$ bytes. The result of query includes 10,000 records assuming each employee is related to department and consider each record in the query is 40 bytes long. The query is submitted at site 3, which is the resultant site as the query result is required here. The original query that extract data from table EMP and table DEP can be executed and implemented in three different ways.

CASE I

Transfer both EMP and DEP relations to the result site perform the join at site 3.

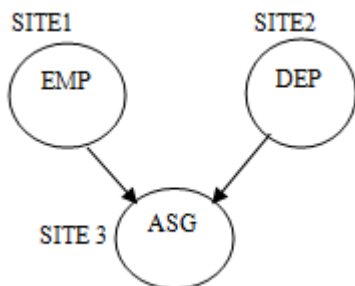


Fig2. Data transfer to SITE 3

The total data transfer is $10,0000 + 3500 = 1003500$ bytes

CASE II

Transfer EMP relation to the site 2, execute the join at site 2 and send the result to site 3.

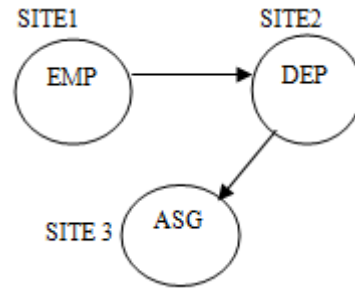


Fig.3 data transfer to SITE2

The size of the total data transfer is $40,0000 + 10000 = 1400000$ bytes. The total data transfer is $40,000 + 3500 = 403500$ bytes.

CASE III

Transfer DEP relation to the result site 1, execute the join at site 1 and send the result to site 3.

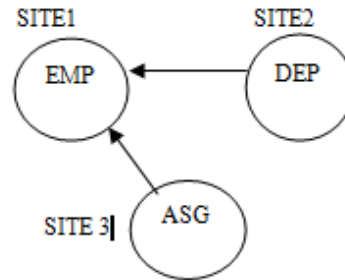


Fig4 data transfer to SITE1

The total data transfer is $40,0000 + 3500 = 403500$ bytes. Strategy 3 is most efficient as no. of bytes transferred is least in this case. A complex strategy which works better than these strategies is semi join[10].

B. Distributed Query Processing using Semi join

The idea behind distributed query processing using the semi join operation is to reduce the number of tuples in a relation before transferring it to another site. The idea is to send the joining column of one relation R to the site where the other relation S is located, this column is then joined with S. Following that, the join attributes, along with the attributes required in the result, are projected out and shipped back to the original site and joined with R. Hence, only the joining column of R is transferred in one direction, and a subset of S with no extraneous tuples or attributes is transferred in the other direction. If only a small fraction of the tuples in S participate in the join, this can be quite an efficient solution to minimizing data transfer. To illustrate this, consider the following strategy for executing Q:

1. Project the join attributes of DEP at site 2 and transfer them at site 1. For Q, there is a transfer of $f = \pi_{DNUMBER}(DEPARTMENT)$ whose size = $4 * 100 = 400$ bytes. Join the transferred file with the EMP relation at site 1 and transfer the required attributes from the resulting file to site 2. For Q, there is a transfer $R =$ whose size = $34 * 10000 = 340000$ bytes.

2. Execute the result by joining the transferred file

Using this strategy, there is a transfer 340,400 bytes for Q . There is limit of the EMPLOYEE attributes and tuples transmitted to site 2 in step 2 . These can send to only those that will actually be joined with a DEPARTMENT tuple in step 3. For query Q, this turned out to include all EMPLOYEE tuples, so little improvement was achieved. The semi join operation was devised to formalize this strategy[10].

5. CONCLUSION AND RESULTS

In distributed database systems data is physically distributed among geographically different locations. when there is need to join the data between sites data has to be transmitted from one site to other. The volume of data is determined using both sub operations join and semi join .we have analyzed that the semi join can reduce the volume of data transmitted more than in case of join as shown in fig.5 . Hence sub operations are decided dynamically in distributed optimizer so that cost can be reduced maximum.

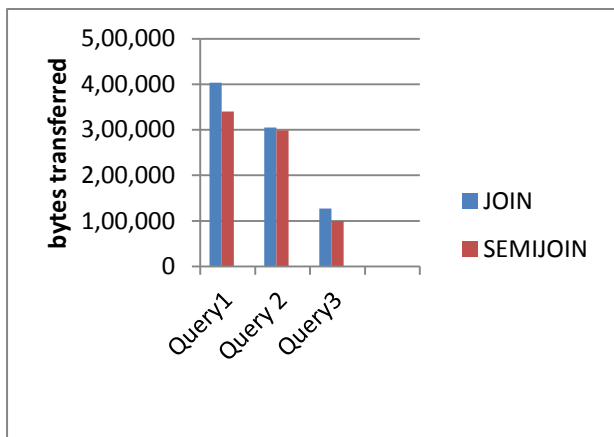


Fig.5 Analysis of data transmission

6. REFERENCES

- [1] Manik Sharma, Dr. Gurvinder singh, Rajinder Virk “Analysis of joins and semi joins in a distributed database query” published in preceding of International journal of computer application (2012)
- [2] Pawandeep Kaur, Jaspreet Kaur Sahiwal “join query optimization in distributed databases” Published in the preceding of International journal of scientific and research publication 2013
- [3] Ridhi Kapoor et al. “selectivity and cost estimates of query optimization in distributed databases“ international journal of enhanced research in management and computer applications (2013)
- [4] M. Tamer Ozsu, Patrick Valduriez, “Principles of Distributed Database Systems”, Third Edition, Springer, 2011
- [5] Ms. Preeti tiwari et al “query optimization strategies in distributed databases” in the proceedings of 2nd international conference on ICETEM 2013
- [6] S. Pramanik, et al. “Optimizing join queries in Distributed Databases” IEEE 2002.
- [7] Sunita M. Mahan, Vaishali P. jadhav “tri-variate optimization strategies of semi join technique o distributed databases” International journal of computer applications (2013)
- [8] Xiaofeng Li et al “ Study of query of distributed database based on relation semi join” International Conference on Computer Design and Applications 2010 .
- [9] Lin Zhou, Yan Chen et. al “The semi join query optimization distributed database system The National Conference on Information Technology and Computer Science 2012.
- [10] “Database Concepts” seventh edition by Navathe.
- [11] Kumar, Saurabh, et al. "Cost-Based Query Optimization with Heuristics." International Journal of Scientific & Engineering Research vol 2.9 2011