Query Optimization in DDBS

Garima Mahajan
M.Tech. Scholar
Guru Nanak Dev University
Amritsar, Punjab India

ABSTRACT

Distributed query processing is an important factor in the overall performance of a distributed database system. The database queries access the applications on the distributed database sites thus the main problem incurred is the minimization of the total operating cost i.e. communication cost and processing. The problem gets more complicated by the fact that query processing not only depends on the operations of the query, but also on the parameter values associated with the query. Query optimization is a difficult task in a distributed client/server environment as data location becomes a major factor. In order to optimize queries accurately, adequate information must be accessible to determine which data access techniques are most effective e.g. table and column cardinality, organization information, and index availability. Optimization algorithms have an important impact on the performance of distributed query processing.

Keywords

Distributed databases, query processing, query optimization

1. INTRODUCTION

A distributed database system is the combination of two different technologies used for data processing: Database Systems and Computer Networks. The main component of a Database is the data which is basically collection of facts about something. This 'something' may be the business data in case of a business corporation, strategic data in case of a military's database etc. Distributed database (DDB) is a collection of multiple, logically interrelated databases distributed over a computer network. [1] [2] Retrieval of data from different sites in a DDB is known as distributed query processing. Distributed processing performs computations on numerous CPUs to attain a single result. [2] In query processing, the database users generally specify what data is required rather than specifying the procedure to retrieve the required data. Thus, an important aspect of query processing is query optimization. Now in query optimization, the optimizer of the database system finds a good way to execute the queries. [3] Query processing is more complex and difficult in distributed environment in comparison to centralized environment as large number of parameters affect the performance of distributed queries, relations may be fragmented and/or replicated, and considering many sites to access, query response time may become very high [2][1] The distributed query optimization has several problems related to the cost model, larger set of queries, optimization cost, and optimization interval. The area of query optimization is very large within the database field.[10] The goal of Distributed Query Processing (DQP) is to execute such queries efficiently in order to minimize the response time and the total communication cost associated with a query. [2]. Therefore it seems logical to look at potential benefits in relation to their costs. This cost/benefit ratio is important for all businesses, but it is especially crucial for smaller ones that do not have the massive resources of Fortune 500 companies [9]. There are three major activities [6] in the processing of distributed database system, in the first phase the database is fragmented, in second phase some complex mechanism is used to allocate the database fragment to the different sites and in the third phase the execution of task takes place. It is believed that an effective database fragmentation improves the performance of the database. No doubt fragmentation increases the complexity of physical database design but it significantly impact performance and manageability [7].

A query normally has many possible execution strategies, and choosing a suitable one for processing a query is known as query optimization. A query is expressed by using a high-level language such as SQL (Structured Query Language) in relational data model.[1] The main function of a relational query processor is to transform a high-level query into an equivalent lower-level query (relational algebra), and the transformation must achieve both correctness and efficiency. Execution strategy for the given query is implemented by lower-level query. Since data is geographically distributed in distributed relational database system, the processing of a distributed query is composed of the following three phases: local processing phase, reduction phase, and final processing phase [4].

- The local processing phase basically involves local processing such as selections(σ) and projections (π).
- The reduction phase uses a sequence of reducers (i.e, semijoins and joins) to reduce the size of relations.
- The final processing phase sends all resulting relations to the assembly site where the final result of the query is constructed. [5]

A straightforward approach of processing a distributed query would involve sending all relations directly to the assembly site, where all joins are performed. The allocation of the data influences the performance of the distributed systems given by the processing time and overall costs required for applications running in the network [8]. In distributed query processing, partitioning a relation into fragments, union of the fragments to form a whole relation, and transferring a relation/fragment from one database to another database are common operations. Both local processing costs and communications costs are taken into account by the optimizers of algorithm R* and Distributed-INGRES. In R*, a join between two relations is performed at a single site by using the nested-loop method or the merge-scan method. For a general query, R* exhaustively enumerates all possible sequences of joins with all possible join methods and allocates joins at each possible site. Since each join is performed at only a single site, existence of multiple processors at different sites is not considered for improving performance through parallel execution. In contrast, Distributed-INGRES uses the "fragment and replicate" query processing strategy. The strategy requires one of the relations referenced by a query to be fragmented and other relations to be replicated at the sites that have a fragment of the fragmented relation. [4]

The structure of the paper is as follows. Section II describes an objective of Distributed Databases. Section III, basically analyses various cost model in distributed databases. We finally concluded the work in the Section VI.

2. OBJECTIVES

An objective of DDBMS is to process distributed queries efficiently and also providing availability and reliability. [1]In a distributed execution environment, we consider two time consumption estimates namely: *total time* or *response time*. [5]

- The Total cost is the sum of the time consumed by each processor, regardless of concurrency.

Local processing time: (CPU + I/O) time

Communication time: fixed time to initiate a message + time to transmit the data

- The response time is the elapsed time between the initiation and the completion of a query[3][2]

Distributed query processing is to translate a high-level query on a single logical distributed database (as seen by the users) into a low-level language on physically distributed local databases. In distributed query processing, the total cost should be minimized for executing a distributed query. [5] The query execution involves only a local processing cost when no relation is fragmented in a distributed DBMS. On the other hand, if relations are fragmented, a communication cost is incurred in addition with the local processing cost. The aim of distributed query processing is to minimize the total execution cost of the query which includes the total processing cost (sum of all local processing costs in participating sites) of the query and the communication cost. The local processing cost of a distributed query is evaluated in terms of the number of disk accesses (I/O cost) and CPU cost. The CPU cost is incurred when performing data operations in the main memory in participating sites. The I/O cost can be minimized by using efficient buffer management technique. In a distributed query execution, the communication cost is required to exchange data between participating sites. Hence, the communication cost depends on several factors such as the amount of data transfer between participating sites, the selection of best site for query execution, the number of message transfer between participating sites, and the communication network. In case of high-speed wide area networks (with a bandwidth few kilobytes per second), the communication cost is the dominant factor and the optimization of CPU cost and I/O cost can be ignored in such cases. The optimization of local processing cost is of greater significance in case of local area networks.[1][2]

3. ANALYSIS OF DISTRIBUTED COST MODEL

Distributed database system, provides data distribution transparency by hiding the data distribution details from the users. [5] Whenever a distributed query is generated at any site of a distributed system, it follows a sequence of phases namely query decomposition, query fragmentation, global query optimization and local query optimization.

The allocation of data considers a set of fragments $= \{f1, f2, \ldots, fn\}$, a set of locations in a network $L = \{l1, l2, \ldots lm\}$, and a set of applications $A = \{a1, a2, \ldots aq\}$ placed at L. These applications need to access the fragments which should be allocated in the locations of a network. The allocation problem consists on finding an optimal distribution of F over L [8]. Thus, distributed cost model includes cost functions to predict the cost of operators, database statistics, base data, and formulas to calculate the sizes of intermediate results.

3.1 Cost Functions

In a distributed system, the cost of processing a query is expressed in terms of the total cost measure or the response time measures .The total cost measure is the sum of all cost components. If no relation is fragmented in the distributed system and the given query includes selection and projection operations, then the total cost measure involves the local processing cost only. However, when join and semijoin operations are executed, communication costs between different sites may be incurred in addition to the local processing cost. Local processing costs are usually evaluated in terms of the number of disk accesses and CPU processing time, while communication costs are expressed in terms of the total amount of data transmitted. [4] For geographically dispersed computer networks, communication cost is normally the dominant consideration, but local processing cost is of greater significance for local networks. But in wide area networks the local processing cost is mostly ignored and an emphasis is made on minimizing the communication cost. [1] Therefore, the total cost measure can be represented by using the following formula.

$$Total\ cost\ measure = T_{cpu} * insts + T_{I/O} * C_0 + C_1 * X$$

where, T_{CPU} is the CPU processing cost per instructions, insts represents the total number of CPU instructions, $T_{I/O}$ is the I/O processing cost per I/O operation, ops represents the total number of I/O operations, C_0 is the start-up cost of initiating transmission, C_1 is a proportionality constant, and X is the amount of data to be transmitted. For wide area networks, the above formula is simplified as follows

$$Total\ cost\ measure = C_0 + C_1 * X$$

The response time measure is the time from the initiation of the query to the time when the answer is produced. The response time measure must consider the parallel local processing costs and the parallel communication costs. A general formula for response time can be expressed as follows.

$$Response\ time\ measure = T_{cpu} * seq\_insts + T_{I/O} * seq\_ops + C_0 + C_1 * seq\_X$$

Where, seq_insts represents the maximum number of CPU instructions that can be performed sequentially, seq_ops represents the maximum number of I/O operations that can be performed sequentially, and seq_X indicates the amount of data that can be transmitted sequentially. If the local processing cost is ignored, then the above formula simplified into

$$Response\ time\ measure = C_0 + C_1 * seq\_X$$

Let us consider that P amount data is to be transmitted from site 1 to site 2, and Q amount of data is to be transmitted from site 3 to site 2 for the execution of a query as shown in fig 3.1

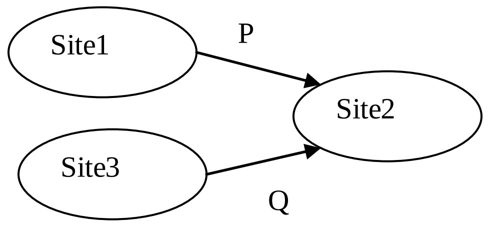

Fig: 3.1 Data Transmission for a Query

Hence, the

$$Total\ cost\ measure = C_0 + C_1 * P + C_0 + C_1 * Q = 2C_0 + C_1$$

Similarly, the

$$Response\ time\ measure = max\ \{ C_0 + C_1 * P, C_0 + C_1 * Q\}$$

because the data transmission is done in parallel. In this case, the local processing cost is ignored. The response time measure can be minimized by increasing the degree of parallel execution, while the total cost measure can be minimized by improving the utilization of resources.[1]

4. CONCLUSION

In this paper we presented an introduction to distributed databases targeting on the two aspects of processing a distributed query namely total cost and response time in distributed databases

5. REFERENCES

[1] Ozsu, M.T., Valduriez,P. Principles of Distributed Database Systems, Pearsons Publications,2010

[2] Ghaemi Reza et al. Evolutionary Query Optimization for Heterogeneous Distributed Database System, 2008

[3]Abdullah, Dilsat. "Query Optimization in Distributed Databases".2003

[4]Monjurul Alom,B.M. "Query Processing and Optimization in Distributed Database Systems". IJCSNS, Vol.9 No.9, Sep 2009.

[5]Bhuyar,P.R. "Horizontal Fragmentation Technique in Distributed Database",(IJSR) International Journal of Scientific and Research Publications, Vol. 2, , May 2012

[6] Rajinder Singh Virk et al., "Optimizing Access Strategies for a Distributed Database Design using Genetic fragmentation", IJCSNS International Journal of Computer Science and Network Security, Vol.11 No.6, Jun 2011

[7] Sharma Manik et al., "A Study of BNP Parallel Task Scheduling Algorithms Metric's for Distributed Database System", IJDPS, Vol.3, No.1, Jan 2012

[8] Morffi ,A.R et al., " A Reinforcement Learning Solution for Allocating Replicated Fragments in a Distributed Database", Computation systems, Vol. 11 No. 2, 2007

[9] Safonov Paul et al., "Is Deployment Of Distributed Processing For Business Applications A Sound Management Decision: Performance Analysis And Simulation", Journal of Information Technology Management Vol.16, No. 3, 2005

[10] Ioannidis, Yannis.E., "Query Optimization", University of Wisconsin, Madison, WI 53706