International Journal of Research in Engineering, Science and Management
Volume-1, Issue-10, October-2018
www.ijresm.com | ISSN (Online): 2581-5782

Query Optimization in Fuzzy Database

Mamta[1], Kapil Kumar[2]
[1]*Intern, Department of IT, Infinite Insights, Delhi, India*
[2]*Assistant Professor, Department of CSE, Sharada University, Delhi, India*

***Abstract*—Query optimization in fuzzy relational databases aims to come out with a minimal execution cost, complexity, time for the available execution strategies. Each type of cost is estimated by these functions. All functions together with their parameters and assumptions forms a model for the fuzzy query optimizer. A function usually takes the size of tables as inputs. There is a possibility that exact data is not available. Also estimating the nature of different models needs to be examined. In this Paper, I have performed the optimization of queries by using the fuzzy data in the database and compared the execution time and cost of queries that is taken by the query to be executed.**

***Index Terms*—Database, Optimization and Query**

I. Introduction

A query is a request for the information from any database. Queries results are generated by accessing relevant database data and manipulating it in a way that results the requested information as query. Since database structures are complex and especially for not-very-simple queries, the needed data for a query can be collected from a database by accessing it in different ways and in different orders. Each different way requires different processing time [10]. Processing times of the same query may have large variance, from a fraction of a second to hours, depending on the way of selection.

II. Query Optimization in Fuzzy Database

The purpose of query optimization, which is an automated process, is to find the way for processing a given query in minimum time. In the query optimization, finding the optimal path to execute the query is complex and costly too and impossible practically. Thus query optimization typically tries to approximate the optimum by comparing several common-sense alternatives to provide in a reasonable time a "good enough" plan [13].

A. Query processing

Query processing requires the transformation of the query into an efficient execution plan. Query optimization requires the best execution plan is the one executed. The main motive is to improve the performance of data by using less resources.Phases of query processing.

1) Phases of query processing

Parsing: At this stage, the syntax (decomposed into a relational algebra and check if it is syntactically correct) and semantics (make sure that tables and columns exist and have permissions to access objects) are checked. At the end parse tree is generated which represent the query's structure.
Optimization: At this stage, query optimizer (simply called as optimizer) present in database as built-in software which determines the most efficient method for a SQL query to access data which is requested. Cost based optimizer (CBO) is used to produce optimal execution plan for query.
Execution: The final stage, to execute the physical query plan that was selected by the CBO.

B. Query Optimization

Query optimization is the overall process of choosing the most efficient way to execute a SQL statement. SQL is non-procedural query language, hence the optimizer is free to do anything like merge, retrieve, recognize, and process the data in any order. The database optimize each SQL statement by analyzing the statistics collected about accessed data. The optimizer focuses to find a way for the efficient optimal plan for a SQL statement by examining several access methods like full table scan and index scans [1]. Fuzzy query optimization framework should provide some techniques like cost estimation and search space for optimal query execution. The basic operations like selection, projection and join are examined for different execution strategies [1].

1) Modes of Optimization

1. The optimizer's job is to find out the efficient optimal execution plan for the query which include DML statements (select, insert, update, delete).
2. CBO uses statistics on table and indexes, the order of tables and columns in SQL statement and available indexes and any user-supplied hints to this to pick the most efficient way to access the requested data. CBO uses the least costly method to get the data.
3. The optimizer assigns different relative numerical cost to each step of that possible plan for a given query, and then sum up all values together to generate an overall cost to estimate for that plan. After calculating costs of all plans, the optimizer chooses the lowest cost plan to estimate. Because of this reason, the optimizer is called as the cost-based optimizer (CBO) [1].
4. CBO almost always perform better than RBO and CBO is preferred and default method. CBO may change the execution plan for same SQL query.
5. RBO use heuristic method to select among several access paths with the help of several rules. All paths are ranked and lowest is chosen.

2) Advantages of query optimization

1. Faster processing of query
2. Less cost per query
3. High performance of the system
4. Less stress on the database
5. Efficient usage of database engine

3) Drawbacks of CBO

1. Execution plans can change across oracle versions, to make sure that same plan is used so need to use stored outlines.
2. The application developer may know the better plan than CBO.
3. CBO depends on current statistics, if the statistics are absent then the optimizer can make poor decisions.

C. Query Optimizer

The optimizer generates the most optimal execution plan for every query block in SQL statement. The optimizer choose the lowest cost plan among all considered plans. The optimizer calculate the overall cost using available statistics. Because the database has so many available internal statistics and tools, the optimizer is in a better position to determine the optimal method of SQL query than the user. Hence all SQL statements use the optimizer [14].

TABLE I
OPTIMIZER OPERATIONS

Operations	Description
Query Transformer	The optimizer checks that the query transformation is required or not. It could help in generating plans.
Estimator	The optimizer estimates the cost of each plan based on statistics in the data dictionary.
Plan Generator	It compares each plan and select the best plan which has lowest cost and it is known as execution plan.

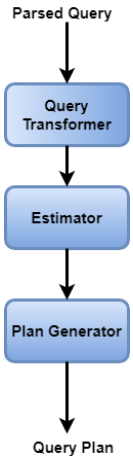

Fig. 1. Optimizer components

D. Execution Plan

A query can have multiple execution plans. The optimizer transforms an SQL query into a set of building blocks called as operators. Each operator performs a basic operation such as scanning an index or filtering rows based on a predicates. Operators are arranged into a query execution plan in the form of a tree, where the root of the tree represents any rows possibly returned by query. An execution plan [11] describes a plan of methods of execution for a SQL query. The plan is the combination of several no. of steps, Oracle Database uses execution plan to execute a SQL statement. Each step either prepares them for the user issuing the statement or retrieves rows of data physically from the database. An execution plan displays assigned cost of the entire plan and each separate operation. The cost is internal unit that is used for the comparison of execution plans. The cost value cannot be changed.

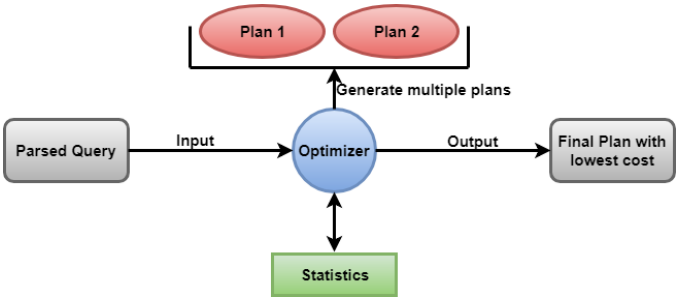

Fig. 2. Executive Plan

1) Query Block and Sub-plans

The input to the optimizer is parsed first for the representation of SQL statement. Query Block is Each SELECT block in the original SQL statement is represented [5]. A query block can be top-level statement, sub-query, or unmerged view. Example:

In the below SQL statement, it has two query blocks:

```
SELECT first_name, last name
FROM hr. Employees
WHERE department_id
IN (SELECT department_id
FROM hr. Departments
WHERE location_id = 1800);
```

The optimizer generates a query sub-plan for each query block. The database optimizes query blocks, each block separately from bottom up direction. The database first optimizes the innermost query block and then generates a sub-plan, and then generates the outer query block which represents the entire query. The number of possible plans for a query block is proportional to number of objects in the FROM clause. This number rises exponentially with number of objects. For example, the possible plans for a join of five tables are significantly higher than the possible plans for join of two tables.

E. Query Optimization Architecture

Two key components of the query evaluation component of database system are the query optimizer and the query execution engine. A set of physical operators is implemented by query execution engine. One or more data streams are taken by the operator as input and it produce an output stream. Examples of physical operators are (external) sort, sequential scan, index scan, nested loop join, and sort-merge join [2]. The

pieces of code are physical operators that are used as building blocks and they help in SQL query execution.

III. Literature Review

In [1], one of the challenges for query Optimization in a multi database system (MDBS) is that some local optimization information may not be accurately known at the global level because of local autonomy. The crisp cost model is not suitable for MDBS because it requires précised data. In this paper, a new technique is presented to perform query optimization.
The following assumptions are made for the MDBS [1]:

1. There are N sites, and all sites are connected to each other.
2. The common global data model is relational; that is, a relational interface is provided for each component DBMS in the MDBS although a component DBMS itself may not be relational.
3. Join strategy, instead of semi-join strategy, is adopted.
4. Joining tables must be on the same site in order for a join to be performed.
5. An n-way join consists of a number of 2-way joins.

The cost of processing a global query in the MDBS consists of the data transmission cost and the local processing cost. Fuzzy Cost Model [1], a cost function is used to estimate each type of costs. A set of cost functions together with their parameters and assumptions forms a cost model. Performance information of a component DBMS or a network are usually reflected in the coefficients of cost functions. In an MDBS, however, such performance information may not be accurately known by the global query optimizer. In this case, fuzzy coefficients (sets) can be used in cost functions to allow imprecise information.
A fuzzy cost model for an MDBS is established to demonstrate that how fuzzy information can be used to estimate the cost of execution strategy for a query. The fuzzy parameters in a fuzzy cost model can be determined and improved by experiments, external characteristics of objects and run time information. It is shown that fuzzy estimated cost usually contains more information than a crisp estimated cost of query. Two reasonable criteria are suggested for the query optimization using a fuzzy cost model. However, the computational complexity of a raw fuzzy query optimization method may be higher than that of the relevant crisp query optimization method.

In [2], Structural Query Language (SQL) is very restrictive in extracting the data from database. Classical SQL queries are capable of data extraction and answer formation from information stored at widely dispersed databases. Human queries are rarely crisp by which it is difficult for the efficient answer formation and data retrieval.. Integration of query languages with fuzzy logic can increase their capability in the data retrieval based on human perception based queries. Query optimization is difficult task in a distributed client/server environment as the data location becomes a major factor. The integration of a query processing subsystem into a distributed database management system with fuzzy logic is used to analyze query response time across fragmentations of global relations. Fuzzy logic based query optimization in distributed databases have an important impact on performance of distributed query processing. Query optimization [2] is the process of producing an optimal or approx. optimal query execution plan which represents an execution strategy for the query. The main task of query optimization is to consider different orderings of operations and minimize total cost associated with execution of requested query. Query optimization is a difficult part of the overall query processing.

TABLE II
Comparison Of Time Complexity for Various Relational Operations

Operations	Time Complexity
Select, Project (with duplicate data)	O(n)
Project (without duplicate data)	O(n log n)
Cartesian Product	O(n2)
Group Join Semi-Join Divisions Set Operators	O(n log n)

They proposed the method of incorporating fuzziness in distributed database for accommodating fuzzy queries which are approximations based on human's knowledge. Fuzzy queries to the relational database are proposed as one candidate model and fuzzy relational database is presented as another relational model. Not only is the data access faster, but a single-point of failure is less occur, and it provides local control of data for users. A distributed database allows the faster local queries and can reduce network traffic. With these benefits comes the issue of maintaining the data integrity [2].

In [9], the problem of evolving databases to make them more intuitive, more user-friendly and to be able to answer fuzzy human queries with separate needs for each user has become a popular research topic. The solution to this problem in part has been proposed via databases that aims at inserting fuzzy data into databases hence handling vague human queries. It has been suggested in many research papers that fuzziness of data may be applied to databases. However, this approach is infeasible and inefficient for real time processing. In the past 30 years of research, fuzzy databases are still not so popular in industry because of unwillingness of companies to replace crisp data with the fuzzy data in their databases due to excessive pre-computation and possible chances of data inconsistency. Having fuzzy databases also places severe constraints on database as it will become very difficult to run crisp queries on the fuzzy databases. This problem is become even more complex with the advent of "Big Data". This proposes fuzzy logic techniques to solve such queries in real time.
Traditional Approach to solve this problem [9]

1. Setup the database and user interface
2. User's all requirements for each attribute
3. Querying the database for each attribute requirement
4. Aggregate results for each attribute

In [3], Relational query language provides a high-level declarative interface to access the data stored in relational databases. Structured Query Language (SQL) is a standard for

storing and retrieving data from relational database. Query optimization is the way to reduce execution time of the query. Query optimizer is a main part in optimizing the process which handles a large input space of complex queries. The query optimization process itself is a complex task. By reducing the execution time, end users get response in short time, but apart from getting response, today's end users are very much interested in specific results based on their inputs. Ranking query or top-k query plays an essential role in retrieving specific information.

Rank Query or Top-k Query [3], Top-k queries are leading in many applications such as web databases, multimedia databases, data mining etc. Rank query gives ability to database system to efficiently retrieval of information based on ranked attribute of user. Top-k query works in applications where users have relatively flexible preferences or specifications for certain attributes. The working of top-K query is an assignment of target values to the particular attributes of a relation. To answer a top-K k query, a database system identifies the objects which match the best according to user specifications, using a given ranking function. For Example: A user is interested in finding a hotel; consider a relation that is hotel in the Mumbai City. There are many tuples in this relation like address, nearest landmark, rent etc. There are three conditions, first is hotel must be near railway station, it should be three stars, and must not be expensive, these three parameters will be considered for ranking. The result to this query is a list of the 10 hotels that match the user's specification closest.

IV. Conclusion and Future Scope

Optimization is much more than transformations and query equivalence in database world. The infrastructure for optimization is very significant. Designing effective and correct SQL transformations is hard, developing robust cost metric is elusive, and building an extensible enumeration architecture is significant undertaking. Despite many years of work, significant open problems remain same. However, an understanding of the existing engineering framework is necessary for making effective contribution to the area of query optimization in database. We plan to further explore other methods to establish a good fuzzy cost model for FDBMS and investigate fuzzy optimization algorithms, for example, fuzzy linear programming, fuzzy dynamic programming, based on a fuzzy cost model. We also plan to develop some heuristics based on fuzzy information to reduce the complexity of fuzzy query optimization.

References

[1] "Establishing a Fuzzy Cost Model for Query Optimization in a Multidatabase System" by Qiang Zhu and Per-Ake Larson in 1994
[2] "Fuzzy Logic Based Query Optimization in Distributed Database" by Abhijeet R Raipurkar and G.R.Bamnote
[3] "An Idea of Extraction of Information Using Query Optimization and Rank Query" by Vivek Shrivastava and Brajesh Patel in 2012
[4] "Analysis of Cost Estimation used in Query Optimization of Fuzzy Relational Databases based on Sort-Merge Algorithm" by S. Deepa in 2015.
[5] "An Overview of Query Optimization in Relational Systems" by Surajit Chaudhuri
[6] "Lotus: A Framework for Query Optimization Based on Distributed Cache" by Chaoyong Li, Gong Cheng,Jinwen Zhong, Can Ma, Weiping Wang, Dan Meng,Qing Wang,Bo Wang in 2015
[7] "Partial Query Optimization Techniques for Partitioned Tables" by Hong Yin, Shuqiang Yang, Hui Zhao, Zhikun Chen in 2012
[8] "An Overview of Query Optimization in Relational Systems" by Surajit Chaudhuri
[9] "Fuzzy Query Processing In Distributed Databases" by Rohan Badlani, Surekha Bhanot and Aditya Mangla
[10] "Query Optimization Using Fuzzy Logic in Integrated Database" by Susana, Suharjito
[11] "A Rule Based View of Query Optimizer" by Johann Christoph Freytag
[12] "The Volcano Optimizer: Extensibility and Efficient Search" by G. Graefe and W.J.McKenna
[13] " The Cascades Framework for Query Optimization" by Goetz Graefe
[14] "A Query Optimization Framework for Fuzzy Relational Databases" by S Deepa.