Distributed Query Processing and Optimization using Fuzzy Logic for Efficient Restaurant Order Management

Adittya Soukarjya Saha [21166056], Mahmudul Haque [20266034]

***Abstract*—Distributed processing of data is a reality and every business wants to use it to stay competitive in the market, especially in the restaurant industry. Even though there are established infrastructures for the distributed data processing, but there are still a number of issues that cause distributed data processing to be depicted as a complex process. Firstly, the state of the distributed systems are bound to change in a rapid manner since the load of sites vary continuously and every now and then new sites get added to the system. Secondly, distributed systems could go up to a massive scale, involving a lot of heterogeneous sites including PCs and mainframe server machines. This paper focuses on query processing for distributed database and information systems and the use of fuzzy logic to further optimize queries for efficient management of orders for the restaurants, thereby tackling the load of sites.**

I. Introduction

In today's world, lot of the restaurant business have been drawn towards the use of distributed databases[4] instead of the centralized database due to some major issues:

- **Scalability and Cost:** A thousand PC processors are now considerably less expensive and powerful than a single large mainframe computer.
- **Different software module integration:** Single product packages sold by a single retailer have a distributed, component-based architecture that allows the vendor to sell and update each component separately.
- **Different branch of business:** Nowadays lots of branches of a single restaurant are distributed over a large span of area, so having distributed databases will enhance the process of order management.
- **New applications:** State of the art applications now make use of distributed database technology and are highly dependent on it.
- **Market forces:** For business to have an edge over the other business, the use of state-of-the-art distributed information technology is seen.

The main target is to have the queries executed as efficiently as possible to reduce the time that the users spend on waiting for the desired answers or the time application programs are delayed.

II. Literature Survey

Preexisting work has already been found on the generalized area of database systems. The research papers mainly reflect the architectures and techniques used for transaction processing [3], followed by studies made on the processing of queries[5], languages, data models[2], and user interfaces for advanced applications. Several properties and priorities are shared by distributed [1] and parallel database systems. The aim of a parallel database system is to enhance response times of queries and transactions as well as the system's availability for applications that are centralized. In this research, we will focus on strategies that are relevant to distributed database systems.

A. S. Saha and M. Haque are with the Department of Computer Science and Engineering, Brac University, 66 Mohakhali, Dhaka 1212, Bangladesh (emails: {adittya.soukarjya.saha, mahmudul.haque}@g.bracu.ac.bd)

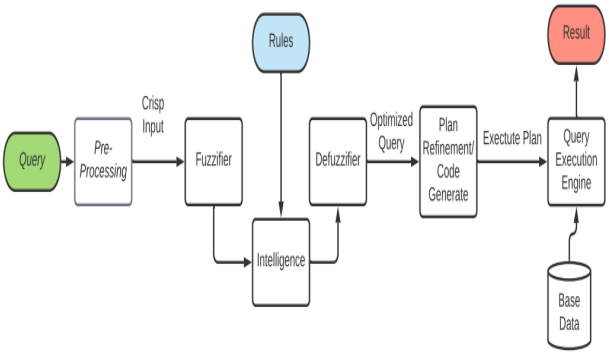


Fig. 1. Phases of query processing

III. Scope of this Paper

In this paper, we will focus primarily on query processing and the optimization of query as shown in **Fig. 1**. Firstly, we will analyze the current condition of the existing database. Then we will try to determine the important variables to use as factors in query optimization, to make the membership function in each set at a predetermined variable, then defining the rules in query optimization. After that, using the fuzzification membership functions that have been made and to do query optimization by inference using Fuzzy Inference System Tsukamoto.

IV. Challenges

Since the standards depend on foreordained guidelines, the dynamic cycle depends on fuzzy logic. On the off chance that these standards are defective, the query optimization may not upgraded. Picking a membership function and fundamental guidelines is perhaps the most difficult pieces of making fuzzy

logic frameworks. Proof of the exactness, consistency and evidence hypothetical culmination of reality span fuzzy logic are either arranged or shown.

V. Conclusion

The test results will be compared to determine the distinction between the reaction time before optimization with fuzzy logic and the reaction time after improvement with fuzzy logic. Correlations are additionally made between the query optimization utilizing fuzzy logic with optimization performance by utilizing a genetic algorithm. The response time of fuzzy logic will in general be a lot quicker contrasted with that of the genetic algorithm utilized for query optimization. Alongside this, the load of sites can be cut down significantly as the queries can be managed quicker.

References

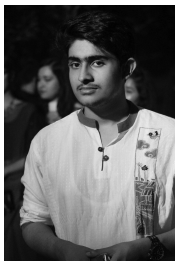
[1] Reinhard Braumandl et al. "Functional-join processing". In: *The VLDB Journal* 8.3 (2000), pp. 156–177.

[2] Peter Buneman and Wang-Chiew Tan. "Provenance in databases". In: *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*. 2007, pp. 1171–1173.

[3] Jim Gray and Andreas Reuter. *Transaction processing: concepts and techniques*. Elsevier, 1992.

[4] Michael Stonebraker. *Readings in database systems*. Morgan Kaufmann Publishers Inc., 1988.

[5] Clement T. Yu and CC Chang. "Distributed query processing". In: *ACM computing surveys (CSUR)* 16.4 (1984), pp. 399–433.



Adittya Soukarjya Saha has completed his B.S. degree with a major in Computer Science and Engineering and a minor in Business Studies from BRAC University, Dhaka, Bangladesh. His research interest includes edge computing, artificial intelligence, quantum computing, reinforcement learning and human computer interaction.

Mahmudul Haque has completed his undergrad degree in Computer Science and Engineering from BRAC University, Dhaka, Bangladesh. His research interest includes edge computing, computer vision, reinforcement learning and quantum computing.