Computing, Information Systems & Development Informatics Journal
Vol. 11 No. 2, 2020 - www.isteams.net/cisdijournal

A Survey of Recent Query Optimization Techniques – A Real Time Sensor Databases Perspective

Enyindah, P., Onyejegbu, L.N. PhD & Onuodu, F.E. PhD
Department of Computer Science
University of Port Harcourt,
Port Harcourt, River State, Nigeria
E-mails: promise.enyindah@uniport.edu.ng; leaticia.onyejegbu@uniport.edu.ng; g.onuodu@gmail.com
Phone: +2349067011341

ABSTRACT

The challenges of query processing in real time sensor databases have brought about some important refinements in the architecturing of query processors dedicated to this purpose. Some of these modifications include amongst others adding multiple processors in a multilayered distributed context, integrating intelligent optimization functions, and using dynamic querying/capturing routines rather than the usual fixed static routines. These modifications also come with certain task specific requirements leading to a variety of database processing requirements. The current theme is moving the databases away from traditional centralized operations to a dynamic and more decentralized or distributed one. The purpose of this survey is to highlight innovative developments in the context of distributed real time query processing in active databases. The paper introduces the relevant background and presents the key researches including dynamic distributed processing, the emergent field of Top-k query processing and optimal query processor in active databases.

Keywords: Active databases, distributed processing, intelligent processing, optimization, query processor

CISDI Journal Reference Format
Enyindah, P., Onyejegbu, L.N. PhD & Onuodu, F.E. PhD (2021): A Survey of Recent Query Optimization Techniques – A Real Time Sensor Databases Perspective. Computing, Information Systems, Development Informatics & Allied Research Journal. Vol 11 No 2, Pp 35-42
DOI - https://doi.org/ 10.22624/AIMS/CISDI/V12N1P5. Available online at www.isteams.net/cisdijournal

1. INTRODUCTION

In traditional systems the query processor functions are often oversimplified and known in advance. This is sufficient as long as data is fixed or static and are not obtained in real time. However, this approach comes with an obvious disadvantage when dealing with dynamic or streaming data content as the state of the data cannot be determined apriori. Distributed Active Database Systems (DADS) represent a possible solution to this serious drawback. Nevertheless, DADS also face some obvious challenges such as poor query execution plans and absence of an improved aggregate query method. This is because; the quality of an execution plan is highly dependent on the accuracy of the estimated number of rows output by each plan operator and also the ability to accept real-time streaming data. Also, the problem of poor query execution plan is responsible for memory pressure in a database, inflated CPU and an overall reduction in concurrency. Some factors that negatively influence query executions include index fragmentation, hidden column correlations, wrong data type assignment/usage, missing data due to query speed issues.

In addition, the mentioned factors that causes poor execution generally affects the query optimizer of the database, which further causes it not to determine the best efficient way to execute a given query task. The purpose of this survey is to highlight pertinent developments in the field of distributed real time sensor database networks which are referred to herein as active databases. This kind of database is useful in two very important application areas:

i. Industrial or commercial field setting such as machine shop scheduling and manufacturing.
ii. Surveillance operations such as in physical-computing (physicom) and cybercrime-detection (cyberdect) systems.

2. RELATED LITERATURE

In this section the researches in the area of real time active databases are presented. Firstly, the background researches following the distributed query processing context is presented. Then the researches in such areas as real time dynamic optimal query processing and Top-k query processing are presented next.

2.1 Related Works in distributed processing context

In **Sanjay et al (2011)**, some issues pertaining replicated data for distributed real-time database systems. The authors provided an overview to compare replication techniques available for these database systems. Data consistency and scalability are the issues that were considered in their work. Those issues are maintaining consistency between the actual state of the real-time object of the external environment and its images as reflected by all its replicas distributed over multiple nodes. Furthermore, the authors discussed a frame to create a replicated real-time database and preserve all timing constrains. In order to enlarge the idea for modelling a large-scale database, they presented a general outline that consider improving the Data consistency and scalability by using an accessible algorithm applied on the both databases, with the goal to lower the degree of replication enables segments to have individual degrees of replication with the purpose of avoiding extreme resource usage, which all together contribute in solving the scalability problem for Distributed Real Time Database Systems.

Farid (2014) researched on a framework of transaction management in distributed Database System Environment. The aim of the work was to determine the transaction management techniques in DDBMS for present time business applications. **Heinz (2014)** proposed a distributed database management system and the data grid for data management in industry setting. The authors outlined both approaches and tried to find commonalities between the two worlds in order to have a most efficient data grid that manages data stored in object-oriented databases. The target object-oriented database management system is Objectivity/DB which is currently the database of choice in some existing High Energy Physics (HEP) experiments as well as in next generation experiments at CERN. **Parul et al (2014)** researched on an overview of distributed databases. The authors presented an overview of Distributed Database System along with their advantages and disadvantages. In addition, they also provided various aspects like replication, fragmentation and various problems that can be faced in distributed database systems.

Katembo (2015) studied a systematic review on distributed databases systems and their techniques. The author proposed a systematic review on distributed databases systems based on respectively three distribution strategies: Data fragmentation, Data allocation and Data replication. **Rai (2015)** investigated an overview of different database security approaches for distributed environment. The main aim of the work was to understand different security problems which have been encountered during designing of a database for distributed environment. Furthermore, the authors tried to give a brief overview about different penetration methods and different security approaches of distributed database to control the security, reliability and privacy of database in distributed environment.

Raymond (2015) focused on distributed database systems. The author reviewed the advantages and disadvantages of distributed database systems, and also discussed issues relevant to their design and use. These issues include concurrency control, distributed query processing and transaction management, disaster recovery, reliability, and methods for distributing data. **Rohan et al (2016)** researched on fuzzy query processing in distributed databases. The study proposed a-three-pronged fuzzy logic-based technique as a layer of computation above traditional query processing to solve such queries in real time. Furthermore, the authors developed a model for improving traditional query processing in order to solve real-time and ambiguous queries. The fuzzy logic-based approach to querying in distributed databases can be used to solve ambiguous queries, incorporating the preferences of each user in the current scenario of excessive data. **Kangseok (2017)** adopted the architecture for scalable, distributed database system built on multicore servers. The author presented performance results for shape similarity queries, which is extremely, time intensive with traditional architectures. According to the work; many scientific fields routinely generate huge datasets. In many cases, these datasets are not static but rapidly grow in size.

3. RELATED WORKS IN REAL TIME DYNAMIC-OPTIMAL QUERY PROCESSING

Tejy et al (2012) investigated a study on optimization techniques and query execution operators that enhances query performance. The work focused on the various Optimization techniques, Components of query optimizer and the Query execution operators that are used for the enhancement of query performance. **Deepali (2013)** proposed a dynamic and randomized query optimization algorithm to improve optimality of access plans. In the study, many different dynamic and randomized query algorithms that compute approximate solutions for producing optimal access plan was studied.

Sirohi (2013) studied an improved database queries for information storage issues. The author provided a case study of a new approach to vastly reduce this CBO (Cost-based Optimizer) dilemma by a combination of denormalized columns and re-writing of the security predicates' sub-queries at run-time, thereby leveling the outer and security sub-queries. The adopted approach by the author results in more stable execution plans in the database and much better performance of such SQLs, effectively leading to higher performance and scalability of the application. **Dhaval et al (2015)** researched on different approaches for query optimization using schema object-base view. In the study, the different techniques were given for optimizing query using schema based and materialized views in data warehouse namely- view adaptation & synchronization, view selection and view maintenance.

Prenner (2015) gave an explanation about query processing, declarative and procedural optimization steps involved and their working with the help of examples. This paper also explains the working of various planners like System-R, SQLite's planner and PostgreSQL's Genetic Planner. The details involved in generating query evaluation plans and estimating them is presented in the paper by Christian. The main emphasis is given on the application of heuristics for optimizer. Use of Pipelining, pushing selection and considering the columns having index on them can eventually help in better query optimization. The paper also explained the fundamental concepts of database query optimization and genetic algorithm. **Jean (2015)** researched on query optimization techniques: tips for writing efficient and faster SQL queries. The study covered how SQL queries can be optimized for better performance.

Shyam (2015) researched on query optimization strategies in distributed databases. According to the work; the query optimization problem in Local Processing Phase: In this phase, the initial large-scale distributed databases is Non-Polynomial-hard (NP-hard) hard in algebraic query specified on global relations is nature and difficult to solve. The complexity of the optimizer increases as the number of relations and transformed in to fragments (Data Decomposition) and made available to the respective sites for processing (Data Localization) like local selections number of joins in a query increase.

Osegi et al (2015) researched on GOptimaEmbed: A Smart Database Management System which uses SMS-SQL for low cost micro-controllers. The authors developed a novel smart database management system (dbms), GOptimaEmbed, for intelligent querying of databases in device constrained embedded systems. The system uses genetic algorithms as main search engine and simplifies the query process using stored in-memory model based on an invented device dependent Short-messaging Structured Query Language, (SMS-SQL) schema translator. In addition, querying is done over the air using integrated GSM module in the smart space. The system has been applied to querying a plant database and results were quite satisfactory. **Legha et al (2016)** presented the Particle Swam Optimization (PSO) algorithm to optimize queries in grid structured databases. They made repetitions on different particles to make the query function on various websites to obtain optimum results which were later compared.

Nathan (2016) proposed a cost-based query optimization via AI planning. The work visited the problem of generating query plans using AI (Artificial Intelligence) automated planning with a view to leveraging significant advances in state-of-the-art planning techniques. The author also focused on the specific problem of cost-based join-order optimization for conjunctive relational queries, a critical component of production-quality query optimizers. He further characterized the general query-planning problem as a delete free problem, and query plan optimization as a context-sensitive cost-optimal planning problem. **Xiao et al (2016)** presented an embedded database query optimization algorithm based on particle swam optimization. the Particle Swam Optimization (PSO) in embedded query processing shown a considerable result in reducing query processing time, but they could not test their proposal for large databases, in large databases it has been found that PSO needs to take care of specific details based on the database to give a good performance.

Yong (2016) reviewed the cougar approach to in-network query processing in sensor networks. The author introduced the cougar approach to tasking sensor networks through declarative queries. Given a user query, a query optimizer generates an efficient query plan for a network query processing. **Asagba (2018)** proposed a query optimization of a distributed database system using fuzzy logic. The author designed a distributed system that optimizes queries and processors, and to develop effective query execution plans for mining distributed database with the use of fuzzy logic. The system was implemented by developing a suitable framework for query processing in a distributed system that is time dependent, and a fuzzy logic approach was employed based on linguistic terminology for flexible user query. The result obtained showed that fuzzy queries will always take less response time as compared to normal SQL queries. As observed, this is due to the fact that query evaluation engine does not have to search all the records in a database, it searches the records within the fuzzy range value alone.

Bennett (2018) researched on hybrid technique for optimization of query processing in a distributed database. In the study, object-oriented design for software development and hardware materials were used. The data shipping techniques was fast in query retrieval as opposed to the query shipping technique however, the had a synergic strength from the two existing techniques to actualize an efficient performance (runtime) Furthermore, the result from the work was similar with previous finding. For server machine, query shipping technique is recommended while for client machine, data shipping technique. **Jeba (2018)** proposed an Improved BAT-based Query Optimization (IBQO) for Crowd-Sourcing System. In the study, preprocessing method was used to mine information from the Crowd. The original population-based ABC algorithm has low convergence speed. Furthermore, the author proposed a novel A-BAT algorithm, which highly improve convergence speed, accuracy and Latency. The algorithm uses a Random walk phase. The proposed algorithm had better optimization accuracy, convergence rate, and robustness.

Sohrabi et al (2018) proposed an Evolutionary Game Theory-based Materialized View Selection (EGTMVS) method where each member of a random population of solutions is considered as a player in the game was presented. The EGMTV have been shown by the authors to outperform other existing techniques by way of query execution time and quality as the number of queries increase and for high volume data. However, it also remains to be seen whether low volume data with streaming characteristic can be handled by the EGTMVS algorithm.

Sharma et al (2018) presented a Clinical Decision Support System (CDSS) query optimizer using a hybrid technique including the Firefly algorithm (FA) and a controlled Genetic Algorithm (GA). In their presented system, using a Restricted Divergence Based Stochastic Clinical Data Query Optimizer (RDFG-CDQO), FLY-A and controlled GA were used to tune the cost function generated and find a better query execution layout from a set of 10 different types of CDSS queries. The results of this technique were compared with four other GA based query optimizers and an exhaustive enumeration optimizer (EEO) technique. It was reported by the authors that their proposed technique will not outperform the existing technique (EEO); however, the authors report that their proposed RDFG-CDQO technique will be better with small and less data-intensive clinical DSS queries.

Iman et al (2018) applied incremental techniques for large-scale dynamic query processing. The work discussed legacy approaches for incremental query processing, and then gave an overview of the new challenges introduced due to processing big data streams. Secondly, they also discussed in detail the recently proposed algorithms that address some of these challenges, and further emphasized the characteristics and algorithmic analysis of various proposed approaches and conclude by discussing future research directions. However, they failed to further implement with a unique query processor.

Zeng et al (2018) presented a modified query technique called M-Skyline Query for the optimal query selection from real and synthetic databases; the Skyline Query (SQ) technique is a popular and very strong multi-criteria, data mining and decision-making tool. However, the authors presented a technique that can handle uncertain databases and in addition considered the sunk cost which was neglected in the original SQ model. Their proposed algorithm comprised three core algorithms – Baseline Algorithm (BA), the Inter-group structure optimization algorithm (IGA) and the Extended Pruning of Groups (EGP). For the real databases analysis, the IGA and the EGP were utilized. The IGP was found to be more efficient than the EGP. The authors however, stressed the need for extending their studies to query processing over incomplete databases and enhancing the approximation accuracy of the reported results to improve speed of processing.

Ryan et al (2019) adopted Neo which is a learned query optimizer. They introduced Neo (Neural Optimizer), a novel learning-based query optimizer that relies on deep neural networks to generate query executions plans. Neo bootstraps its query optimization model from existing optimizers and continues to learn from incoming queries, building upon its successes and learning from its failures. Furthermore, Neo naturally adapts to underlying data patterns and is robust to estimation errors. Experimental results demonstrate that Neo, even when bootstrapped from a simple optimizer like PostgreSQL, can learn a model that offers similar performance to state-of-the-art commercial optimizers, and in some cases even surpass them.

4. RELATED WORKS IN TOP-K QUERY PROCESSING

Kakad et al (2013) developed a system for improved query processing in distributed databases involving the use of an arbitrary Top-k Query (TkQ) max-value filtering technique. In their proposed system the child-parent (client-server) topology was used for the processing and coordination of Top-k queries respectively. They used the median-filter paradigm as a basis for max query value identification and control suitable for emergencies or critical infrastructures in industrial setting. They compared their proposed solution to that without filtering and showed substantial improvements in Top-k query value handling.

Ihab (2016) proposed a Supporting Top-k join Queries in Relational Databases. The authors addressed top-k join queries in relational query processors. They also introduced a new rank-join algorithm that makes use of the individual orders of its inputs to produce join results ordered on a user-specified scoring function. The idea is to rank the join results progressively during the join operation. Furthermore, they introduced two physical query operators based on variants of ripple join that implement the rank-join algorithm.

5. CONCLUSIONS AND RECOMMENDATIONS FOR FURTHER STUDIES

The future of active databases is dependent on the effectiveness of existing strategies. These strategies have the obvious requirements of optimality, distributed capabilities and dynamicity. Thus, a modern day state-of-the-art query processor dedicated to an active database must be adaptive in the real world sense. It has to be able handle data that is of a streaming nature and inconsistencies in application or task specific requirements.

This survey has presented an overview of existing or related researches and techniques in the field of distributed active database systems (DADS) suitable for real world deployments. The survey categorized these researches broadly into three categories bordering on distributed, dynamic-optimal and Top-k query processing respectively. From the reviewed researches, it is obvious there is a need to improve on the existing state-of-the-art. Thus, the novelty in existing research can be exploited to make them capable for real industry world setting.

In this regard, the following recommendations are made:

i. The deployment of chains of multiple real time sensors for point-signal data acquisition. This will support the idea of redundancy in ensuring reliable data estimation and hence reporting.
ii. The development of swarming query processors. This will support the idea of dynamic processing and multi-agent best search during Top-k query processing.
iii. The use of No-sql database systems. This will facilitate dynamic querying to be made in near real time using traditional centralized tools.

Furthermore, we recommend intelligent swarming agent based query processing for DADs within an industrial and commercial setting or within real time surveillance infrastructures.

REFERENCES

1. Asagba P.O. (2018), Query Optimization of a Distributed Database System using Fuzzy Logic, *International Journal of Engineering Science Invention (IJESI), 7(1), 67 – 74*.
2. Bennett O. (2018), Hybrid Technique for Optimization of Query Processing in a Distributed Database, *International Journal of Computer Science and Mathematical Theory (IJCSMT), 4(2), 19 - 27*.
3. Deepali A. (2013), Dynamic and Randomized Query Optimization Algorithms to Improve Optimality of Access Plans, *International Journal of Scientific and Engineering Research*
4. Dhaval P., and P. Patel (2015), A review paper on different approaches for query optimization using schema object base view, *International Journal of Computer Application (IJCA), 114(4), 16 – 18*.
5. Farid A. (2014), A Framework of Transaction Management in Distributed Database System Environment. *International Journal of Advanced Research in IT and Engineering (IJARIE), 3(2), 35 – 53.*
6. Heinz S. (2014), Distributed Database Management Systems and the Data Grid, *Journal of Computer Science and Informatics (JCSI), 4(7), 21 – 28*.
7. Ihab I. (2016), Supporting Top-K Join Queries in Relational Databases, *International Journal of Engineering Technology (IJET), 4(7), 131 – 137.*
8. Iman E. (2018), Incremental Techniques for Large-Scale Dynamic Query Processing, 27th ACM *International Conference on Information and Knowledge Management (CIKM, 18), 67 – 73.*
9. Jean H. (2015), Query Optimization Techniques: Tips for writing efficient and faster SQL queries, *International Journal of Scientific and Technology Research (IJSTR), 4(10), 22-26.*
10. Jeba J.R. (2018), An Improved BAT-Based Query Optimization for Crowd-Sourcing System, *International Journal of Intelligent Systems and Applications 3(7), 33 – 40.*
11. Prenner, J. A. (2015, February). An Introduction to Query Optimization in Relational Databases. In *Seminar in Data and Knowledge Engineering*.
12. Kakad, S., Sarode, P., & Bakal, J. W. (2013). Analysis and Implementation of Top K Query Response Time Optimization Approach for Reliable Data Communication in Wireless Sensor Network. *IJEIT*, 3.
13. Kangseok K. (2017), Architecture for Scalable, Distributed Database System built on Multicore Servers, *International Journal of Computer Application (IJCA), 8(19), 14 – 19.*
14. Katembo K. (2015), A Systematic Review on Distributed Databases System and their Techniques. *Journal of Theoretical and Applied Information Technology (IJTAIT) 96(1), 1-31*.
15. Legha, M. M., & Afsharmanesh, M. (2016). Demand Management in a Microgrid Using Particle Swarm Optimization (PSO). *Specialty journal of Engineering and Applied Science*, *1*(1), 47-55.
16. Nathan R. (2016), Cost-based Query Optimization via AI planning, *International Journal of Computer Application (IJCA), 4(7), 18 – 27.*
17. Osegi, N. E., & Enyindah, P. (2015). GOEmbed: A Smart SMS-SQL Database Management System for Low-Cost Microcontrollers. *African Journal of Computing & ICT*, *8*(2), 133-144.
18. Parul T. and I. Megha (2014), An Overview of Distributed Databases, *International Journal of Information and Computation Technology, 4(2), 207 – 214.*
19. Rai P.K. (2015), An Overview of Different Database Security Approaches for Distributed Environment, *International Journal of Innovative Science, Engineering and Technology (IJISET), 2(6), 316 – 321*.
20. Raymond B. (2015), Distributed Database Systems, *International Journal of Computer Applications (IJCA), 5(3), 1-8.*
21. Rohan B, Bhanot S, Mangla A. (2016), Fuzzy Query Processing in Distributed Databases, *International Journal of Advanced Computational Engineering and Networking, 4(1), 68–73.*
22. Ryan M. (2019), Neo: A learned Query Optimizer, arxiv:1904.03711 v 1[cs.db], *https://arxiv.org/pdf/ 1904.03711.pdf, 23 -29.*

23. Sanjay K., K.Sharma, S.Vishnu (2011), Issues in Replicated Data for Distributed Real-time Database Systems, *International Journal of Computer Science and Information Technologies (IJCSIT), 2(4), 1364 – 1371.*
24. Sharma, M., Singh, G., & Singh, R. (2018, in-press). Clinical decision support system query optimizer using hybrid Firefly and controlled Genetic Algorithm. *Journal of King Saud University-Computer and Information Sciences*, 5(6), *22-28*.
25. Sohrabi, M. K., & Azgomi, H. (2018, in-press). Evolutionary game theory approach to materialized view selection in data warehouses. *Knowledge-Based Systems, pp 44-47*.
26. Shyam P. (2015), Query optimization strategies in distributed databases, *International Journal of Computer Science and Information Technologies, 6(5), 4228 - 4234*
27. Sirohi A.K. (2013), Improved Database Queries for Information Storage Issues, *International Journal of Database Management Systems (IJDMS), 5(1), 1 - 16*
28. Tejy J., and S.K. Srivatsa (2012), A study on optimization technique and query execution operators that enhances query performance, *International Journal of Advanced Research in Computer Science (IJARCS), 3(3), 228 – 233*.
29. Xiao Mingyao, Lixongtei. 2017. Embedded Database Query Optimization Algorithm Based on Particle swarm. *Seventh international conference on measuring Technology 1-14*.
30. Yong Y. (2016), The Cougar Approach to In-Network Query Processing in Sensor Networks, *International Journal of Computer Applications (IJCA), 10(9), 44 – 49*.
31. Zeng, Y., Chen, G., Li, K., Zhou, Y., Zhou, X., & Li, K. (2018, in-press). M-Skyline: Taking sunk cost and alternative recommendation in consideration for skyline query on uncertain data. *Knowledge-Based Systems, 45-49*.