International Journal of Engineering Inventions
e-ISSN: 2278-7461, p-ISSN: 2319-6491
Volume 12, Issue 9 [September. 2023] PP: 120-126

Investigating the Performance of Query Optimization & Parallel Processing Techniques in Information Systems

Yousef Rahimy Akhondzadeh
Department of Computer and Information Technology Engineering, Qazvin Branch, Islamic Azad University, Qazvin, IRAN
Corresponding Author: Yousef Rahimy Akhondzadeh

ABSTRACT: *Today, due to the rapid development of technology and at the same time, the rapid growth of information, and also, considering that in current computing programs, storing information, processing and maintaining results has become one of the daily and challenging tasks of database managers, discussions such as parallelization and simultaneous execution of tasks and how to optimize and execute complex queries with specific objectives in a time slice have become one of the important topics in the field of processing systems. Parallel processing and multi-processing is one of the topics that is discussed today in almost all fields of information technology, including database management systems, data mining, distributed data processing, data warehouses, etc. Therefore, reviewing the work done in this field and providing new solutions, optimizing and modifying existing algorithms can play an important role in increasing the speed and efficiency of these systems.*

Date of Submission: 07-09-2023 Date of acceptance: 20-09-2023

I. INTRODUCTION (PROBLEM STATEMENT)

A traditional computer usually has a single processor that is suitable for executing one task. One of the ways to increase the speed of calculations is to use several processors in a single computer (multi-processor) or instead to use several computers with a single processor. One of the ways to increase the speed of calculations is to use several processors in a single computer (multi-processor) or instead to use several computers with a single processor. Also, depending on the size of the problem and its degree of parallelism, they work on a single problem, and with the continuous improvement of the execution speed, it enables real-time parallel calculations [1].

To extract applicable and useful information from big data, today's IT world is facing big problems. Currently, the available data is in terms of terabytes or petabytes, and in the near future, the volume of this data will increase to petabytes or zettabytes [1]. In most current computing systems, storing and processing information and maintaining results intermittently has become one of the challenging and daily tasks for database administrators. Therefore, efforts have been made to parallelize and execute tasks at the same time, or how to optimize and execute complex queries with specific objectives in a time slice. Therefore, the issues of parallelization and the use of database management systems with the ability to parallelize and process large data have been raised and received more attention.

It is predicted that in 2025, the number of users who have access to the Internet will be 75% of the total population of the planet [3]. They will include groups such as young children, the elderly, and residents of developing countries – all who previously had little or no Internet use. This not only leads to an increase in the volume of data, but also leads to a very sudden increase in the amount of processing and requires extensive changes in the collection and processing of mobile phone data and real-time information. Also, by 2025, more than 20% of generated data will be accumulated in real time [18]. Storing, managing and processing such data requires the use of a new generation of database management systems with special capabilities.

One of the capabilities of any database management system is the execution of various types of queries. These queries can be queries with single relationships or queries with multiple relationships and various operations. Most of these queries usually have a number of relational operators such as JOIN types, aggregations, etc. (single or multiple). Since the existing parallel processing methods that handle these queries are very sensitive to how the data is stored, they include more IO costs in their processing [11]. Also, in cases where the data is massive and distributed, in addition to IO costs, communication costs should also be considered.

II. RESEARCH OBJECTIVES & METHODS

In this article, the concepts related to parallel processing, multi-processing and multi-processor techniques are discussed, as well as the work that researchers have done in this field so far. Also, cases of parallelization applications in data mining, distributed data systems, and massive database management systems, as well as the combination of query optimization techniques with parallel processing techniques, etc., are also examined. Finally, the advantages of using parallel processing techniques in speeding up processing operations and managing large data are discussed.

III. WHAT IS PARALLEL PROCESSING & WHY?

Parallel processing is a method of handling or doing multiple parts of a task by two or more processors. Separating different parts of a task among multiple processors greatly reduces the execution time of a program. Any system with more than one processor, as well as the multi-core processors commonly found in computers today, can perform parallel processing [17].

Multi-core processors are chips that contain two or more processors for better performance, lower power consumption, and more efficient processing of multiple tasks. This multi-core structure is almost similar to installing several separate processors in a computer. Most common computers may have two to four cores (can be increased to 12 cores) [17].

Parallel processing is usually used to perform complex tasks and calculations. Data scientists usually use parallel processing for computational tasks and intensive and voluminous data [17].

Typically, a computer scientist divides a complex task into several parts with a software tool and assigns each part to a processor. Each processor then solves its own part, and the data generated is aggregated by another software tool to provide a solution or re-run the task. In this structure, each processor does its normal work, and at the same time, the combination of processors does the work in parallel and according to the command, and pulls the data out of the computer's memory. Processors also rely on special software to communicate with each other so that they can be in sync with each other about changes in data values. Assuming that all the processors are in sync with each other, at the end of the job, the software should put all the pieces of data together. Also, computers without multiple processors (single processor) can also be used in parallel processing, if they are networked together and form a cluster (multiprocessing technique).

There are different types of parallel processing, the two most common of which are SIMD (Single Instruction, Multiple Data) and MIMD (Multiple Instruction, Multiple Data). SIMD, is a form of parallel processing in which a computer system has two or more processors with the same instruction set, while each processor handles different data. SIMD is commonly used to analyze large data sets that are identical based on specified criteria. MIMD, is another common form of parallel processing where any computer system with two or more processors receives data from separate data streams. There is also another type of parallel processing called MISD or Multiple Instruction Single Data, which is less commonly used, in which each processor uses a different algorithm for the same input data [17].

IV. PARALLELIZATION IN DATA MINING

A group of researchers have used parallelization and parallel processing techniques in data mining problems. Therefore, the use of parallel processors and the implementation of different algorithms make association rules always a positive point [1].

Extracting association rules in large databases is a challenging task. An Apriori algorithm is widely used to find sets of duplicate items in a large database. But this algorithm will be inefficient in case of a normal database because it requires more I/O load. Most of the problems of this algorithm have been solved by many other algorithms and even parallel algorithms, but these algorithms are also inefficient to find sets of repeated items from large databases with less time and more efficiency. Therefore, a hybrid architecture is proposed, which consists of integrated parallel and distributed computing concepts. The main idea of this architecture is to combine distributed and parallel computing in such a way that it is efficient to find sets of repeated items from a large database in less time [1].

An association rule plays an important role in data mining and marketing techniques. Buying a product together with other related products indicates an association rule. Association rules are used to show relationships between data items. Association rules are often used for various purposes such as marketing, advertising, and target markets, and find common uses of items. Such issues are detected by applications and developed as shopping cart analysis to find relationships between items purchased by customers. For example, what types of products tend to be purchased together? [1].

Association rule extraction algorithms play a major role in data mining research. These types of algorithms are one of the types of data mining algorithms that are used to find the relationship between the items of a set. The extraction of association rules is not only widely used to find repeating patterns, but also can be developed for repeating pattern prediction. Their main drawback is that it takes a lot of computing time, so in

order to reduce the execution time, other parallel and distribution-based algorithms are also introduced [1].

Data mining concepts and techniques are applicable in all fields of application. Their application has also been extended from market portfolio analysis to a variety of areas such as customer classification, doctors, e-commerce, classification and clustering. To improve the efficiency of mining algorithms, parallelization is considered as an effective concept. Although there are some application areas in which we cannot exploit parallelization successfully, but some computing programs can be easily parallelized. However, most algorithms suffer from their time-consuming processes. Parallel and distributed algorithms solve this problem by using parallelization techniques [1].

V. PARALLELIZATION & DISTRIBUTED DATA

Existing association rule extraction algorithms and modules are deployed on a centralized environment such as massive databases or data warehouses. With the development of distributed databases and communication technologies, they do not meet the needs of extracting association rules from distributed datasets [1]. For this reason, to achieve high-performance parallel computing, it is necessary to reduce redundant computations and avoid excessive communication between parallel tasks and achieve load balance. In this regard, YueShen and Zhaongqian Fu propose a parallel algorithm based on a multiprocessor, which is related to the distribution of parallel tasks to improve the use of computing resources. These methods often require compromises in computation, communication, load balancing, etc. This method does not require additional communication or computation, but can achieve load balancing to make full use of computing resources [1].

The largeness and high dimensionality of data sets that are typically available as input to an associative rule extraction problem make them an ideal problem to solve on multiple processors in parallel. Its main reasons are the limitations of memory and processor speed that single processor systems face [1].

Developing fast and efficient algorithms that can handle large amounts of data becomes a challenging task due to the existence of large databases. N. Li et al. implement a parallel Apriori algorithm based on MapReduce, which is a framework for processing large datasets on certain types of distributed problems using a large number of computers. This proposed algorithm can effectively and efficiently process large data sets on the relevant hardware [9].

In emerging networked environments, we encounter situations where databases residing in geographically distinct sites must collaborate to analyze their data together. But due to the huge volume of data sets, transferring large data sets from across the network to shared servers is neither feasible nor safe. These new algorithms can process databases in their own places and by exchanging the required information between them and obtain the same results as if the databases were merged [12]. S. Bhatnagar presents an algorithm for extracting association rules from distributed databases by exchanging only the required data summaries among them [1].

VI. PARALLEL PROCESSING & MULTI-PROCESSING

We know that the lack of existing algorithms or systems is the main reason for problems such as dealing with large amounts of data, communication slag, increasing speed, determining the support threshold and multiple database scans. Parallel and distributed algorithms using parallelization techniques solve all these problems in static data mining and have made it easy to obtain sets of repeated items with better temporal and spatial complexity [1].

A traditional computer has a single processor to perform only one task. One of the ways to increase the speed of calculations is to use the multi-processing technique. It means to parallelize several single-processor systems in such a way that they can work on the same problem at the same time. Another way to increase the speed of calculations is to use multiple processors. It means to use systems that have several processors. There is another proposed system that uses both methods. In fact, these types of systems result from the combination of several systems, each of which in turn has several processors. In this scenario, the general idea is to divide the database into a number of clusters and reduce the number of candidates to fit in the main memory easily, even if the database is large. Therefore, to reduce the number of candidates, we divide the entire database into different clusters using the PAFI algorithm. After finding the clusters, the transaction reduction matrix method is applied to each cluster so that we don't need to scan the database again [8].

Master-Slave configuration is used in the above proposed architecture. Where the master processor divides the entire database into different clusters and distributes the clusters among the slave processors [16]. The slave processor generates the repeated itemset using the parallel system concept and sends the repeated itemset to the master processor. As shown in Figure 1, the proposed system uses two algorithms. Initially, in the main processor, PAFI (Partition Algorithm Frequent Items) is used for clustering, and in the slave system, the Matrix method is used in each cluster [1].

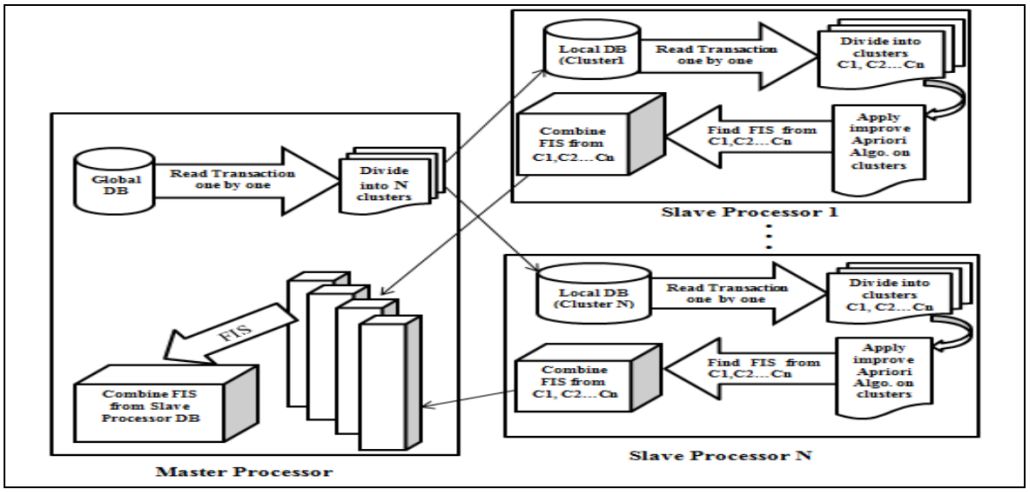

Fig. 1 General architecture flow of the proposed Master-Slave system

VII. PARALLEL PROCESSING & OPTIMIZATION OF QUERIES

Parallel processing and query optimization are two critical techniques used in database management. Parallel processing involves dividing large tasks into smaller subtasks that run simultaneously on multiple processors, while query optimization aims to improve the performance of a database system by finding the most efficient way to execute a query. These techniques are closely related and their combination can significantly reduce the time required to process complex queries on large datasets [2].

In the context of databases, parallel processing involves dividing a query into smaller subqueries and executing them simultaneously on multiple processors. This approach can significantly reduce the time required to execute a query, especially for large and complex data sets [10][13].

The purpose of query optimization is to find the most efficient way to execute the query considering the available resources and database structure. This includes query analysis, evaluation of different execution plans, and selection of the most efficient ones [7][14][15].

Parallel processing and query optimization are closely related because parallel processing can also be used to improve the performance of query optimization. Query optimization algorithms can evaluate different execution plans simultaneously and select the most efficient ones by executing multiple sub-queries in parallel. This can significantly improve the performance of database systems, especially for queries involving complex operations and large data sets. Parallel processing and query optimization are two important techniques to improve the performance and efficiency of database systems. By using parallel processing techniques to execute multiple subqueries simultaneously and optimizing the execution plan of each subquery, database administrators can significantly reduce the time required to process complex queries and improve overall system performance. As databases continue to grow in size and complexity, parallel processing and query optimization are becoming increasingly important to ensure that they remain efficient and effective tools for data management [2].

Generally, query processing is a vital component of many applications in various industries. Their ability to extract insights from massive data sets has made them an indispensable tool in business intelligence, e-commerce, healthcare, social media, and finance, among other fields. As datasets continue to grow in size and complexity, so does the importance of efficient and effective query processing [2].

Query processing using parallel computing is a technique that involves dividing a large query into smaller subqueries and executing them simultaneously on multiple processors. Parallel computing can significantly improve query processing performance by reducing the time required to execute complex queries on large datasets [2].

Another way to increase the speed of query execution is to optimize them. The DBMS optimizer prepares a plan on how to perform a query in parallel, which is done based on the settings of the database manager and based on the resources of the desired server [11].

If our database server is SQL, it can perform a query or index operation in parallel using several operating system threads, this operation can be done with high speed and efficiency. During query optimization, SQL Server looks for queries or indexing operations that can benefit from parallel execution. For these queries, SQL Server inserts a series of exchange operators into the query execution plan to prepare the query for parallel execution. An exchange operator is an operator in the query execution plan that provides process management, data redistribution, and data flow control. After inserting exchange operators, the result of a query execution plan is parallel. A parallel query execution plan can use more than one thread while a serial execution plan used by a non-parallel query uses only one thread for execution. The actual number of threads used by a parallel

query is determined at the very beginning of the execution of the query plan based on the complexity of the plan and the level of parallelism. The degree of parallelism is equal to the maximum number of CPUs that are in use and its values are set at the server level. This can be done using a stored procedure called sp_configure [11].
The query processing process using parallel computing includes the following steps [18]:

- **Query Optimization:** Before executing a query, the query optimizer analyzes it and creates an execution plan that specifies the steps required to retrieve the data. The optimizer can use parallel processing to generate multiple execution plans, each of which can be executed simultaneously on different processors [18].
- **Query Partitioning:** The query partitioner divides it into smaller subqueries that can be executed independently on different processors. The partitioning strategy can be different depending on the data structure and available computing resources [18].
- **Data Distribution:** The data distributor allocates subsets of data to each processor. Depending on the nature of the query and data distribution, data can be distributed in different ways such as row, column or block [18].
- **Query Execution:** Subqueries are executed simultaneously on different processors, with each sub-processor processing its assigned subset of data. Then the results of the sub-queries are combined to create the final query result [18].

Advantages of query processing using parallel computing [18]:

- **Performance improvement:** Parallel computing can significantly reduce the time required to execute complex queries on large datasets, resulting in faster query processing times [18].
- **Increased scalability:** Parallel computing can enable database systems to handle larger volumes of data and more complex queries without significantly increasing processing time [18].
- **Enhanced fault tolerance:** Parallel computing can improve fault tolerance by allowing the system to continue processing queries even if one or more processors fail [18].

The reasons for the need for query optimization and parallel processing [6][18]:

- **Improved performance:** Query optimization and parallel processing can significantly reduce the time required to execute complex queries on large datasets. By dividing queries into smaller subqueries and executing them simultaneously on multiple processors, parallel processing can significantly reduce query processing time. Meanwhile, query optimization can improve query performance by creating an optimal execution plan that minimizes the amount of data to be processed [6][18].
- **Increased scalability:** Query optimization and parallel processing can enable database systems to handle larger volumes of data and more complex queries without significantly increasing processing time. This can be especially important for businesses that need to process large amounts of data in real time or near real time.
- **Enhanced fault tolerance:** Parallel processing can improve fault tolerance by allowing the system to continue processing queries even if one or more processors fail. This can help prevent data loss and ensure that critical business processes work properly [6][18].
- **Cost reduction:** Query optimization and parallel processing can help reduce the cost of running database systems by minimizing the amount of hardware required to process large data sets. Parallel processing by distributing the workload across multiple processors can reduce the need for expensive hardware upgrades [6][18].

VIII. PARALLEL PROCESSING & DATA WAREHOUSES

In the computer world, a data warehouse is a database for storing a history of data. Data warehouse can also refer to a set of hardware and software components that can be used to achieve better analysis of large amounts of data to make better decisions. They can also be used to understand business trends and predict better decision-making, as well as analyze daily sales information and make quick decisions affecting a company's performance. The data warehouse system needs to process a large amount of data, this system is usually used using SMP (Symmetric Multiprocessing) and MPP (Massively Parallel Processing) processing technology. The main structure of the data warehouse includes the input and output of data from the processes, which can be divided into four layers, data source, data staging area, data presentation area and data access tools [4].

MPP provides a cost-effective environment in a data warehouse environment that allows companies to take advantage of maintenance cost performance. MPP refers to systems that have two or more processors that work together to perform an operation, and each processor includes its own memory, operating system, and disks. MPP provides an analytics Data Warehouse platform for data discovery and analysis of large volumes of data [4].

IX. AFFECTING FACTORS OF A DATABASE PERFORMANCE

- **System Resources:** Database performance is highly dependent on disk I/O and memory usage. To fine-tune performance expectations, you need to know how the hardware of your DBMS is running works. The performance of hardware components such as CPUs, hard drives, disk controllers, RAM, and network interfaces significantly affect the performance speed of your database [6][18].
- **Workload:** Workload is equal to the total demand of the DBMS and changes over time. The total workload is a combination of user queries, programs, batch jobs, transactions, and system commands that are routed through the DBMS at a given time [6][18].
- **Throughput:** The throughput of a system defines its overall ability to process data. DBMS throughput is measured in terms of queries per second, transactions per second, or average response time. DBMS throughput is closely related to the processing capacity of the underlying systems (I/O of disks, processor speed, bandwidth, memory, etc.), so it is important to know the throughput capacity of your hardware when determining DBMS operational goals.
- **Contention:** Contention is a situation in which two or more workload components try to use the system in a conflicting way. For instance, multiple queries trying to update a single piece of data at a same time, or multiple large workloads competing for system resources. As the number of collisions increases, the throughput also decreases [6][18].
- **Optimization:** DBMS optimization can affect the overall performance of the system. Database configuration parameters, table design, data distribution, etc. enable the database query optimizer to create the most efficient access schemes [6][18].

X. SOLUTION PROPOSED BY RESEARCHERS

In the case of relational databases, the minimum set of steps to speed up data processing is as follows:

- Upgrading the database schema is the first step that should be taken into account. It is recommended [5] to first normalize the data schema and then partially denormalize it according to the query forms. Database schema normalization may be applied as one of the traditional methods, based on the characteristics of functional dependencies and methods based on graph theory and lists [18].
- Upgrade database queries. In this case, the degree of frequency or usage of the query only affects the way it is stored in the database. If this is a query that runs regularly, it's best to modify it once and save it to the database. In all respects, all queries must go through the following steps: (I) creating a parallel query plan, (II) optimizing the parallel query plan, and (III) optimizing the successive parts of the plan corresponding to the branches of the parallel plan [18].

XI. CONCLUSION

Studies have shown that all information graph optimization methods for various parameters (time, computing nodes, transfer rate between processors, etc.) developed for classical algorithms are also quite suitable for queries [19]. But, despite all these features and good capabilities of these methods, queries in relational databases have their own characteristics, and therefore direct use of the mentioned methods will only speed things up in the best case, and will have no effect in the worst case. This is primarily because the input data for the queries are full tables and they contain a large amount of data. Second, information graphs only provide information about task parallelism, and queries that contain a large number of independent subqueries are rarely encountered in practice. Therefore, often in information graphs in the first approximation, the internal parallelism is zero [19].

REFRENCES

[1]. Anil Vasoya, Dr. Nitin Koli, [2016] "Mining of association rules on large database using distributed and parallel computing", 7th International Conference on Communication, Computing and Virtualization 2016, 79 (2016) 221-230.

[2]. Ankur Bhardwaj, [2022] "Exploring the Synergy between Query Optimization and Parallel Processing for Efficient Database Management", International Journal of Computer Applications & Information Technology, Vol.13, No.2, Jun-Dec, 2022.

[3]. Apache Parquet, Apache Parquet, Internet: https://parquet.apache.org.

[4]. Fajar Ciputra Daeng Bani, Suharjito, Diana, Abba Suganda Girsang, [2018] "Implementation of Database Massively Parallel Processing System to Build Scalability on Process Data Warehouse", 3rd International Conference on Computer Science and Computational Intelligence 2018, 135 (2018) 68-79.

[5]. G.Eadon, E.I.Chong, S.Shankar, A.Raghavan, J.Srinivasan, S.Das, [2008] "Supporting table partitioning by reference in Oracle", in: Proceedings of the ACM International Conference Special Interest Group on Management of Data (SIGMOD), 1111–1122, Vancouver, Canada, June 9–12, 2008.

[6]. Ganguly, Sumit, Waqar Hasan, and Ravi Krishnamurthy, [1992] "Query optimization for parallel execution", Proceedings of the 1992 ACM SIGMOD international conference on management of data. 1992.

[7]. Ioannidis, Yannis E. [1996] "Query optimization", ACM Computing Surveys (CSUR) 28.1 (1996): 121-123.

[8]. K.Belbachir, H.Belbachir, [2012] "The Parallelization of Algorithm Based on Partition Principle for Association Rules Discovery", IEEE International Conference in Multimedia Computing and Systems (ICMCS), pp.1-6, 2012.

[9]. K.Shah and S.Mahajan, [2009] "Maximizing the Efficiency of ParallelApriori Algorithm", IEEE International Conference on Advances in Recent Technologies in Communication and Computing, pp.107-109, 2009.
[10]. Moldovan, Dan I. "Parallel processing from applications to systems", Elsevier, eBook ISBN: 9781483297514, 2014.
[11]. P.Mohankumar, P.Kumaresan, Dr. J.Vaideeswaran, [2011] "Optimism analysis of parallel queries in databases through multicores", Internationa Journal of Database Management System (IJDMS), Vol.3, No.1, February 2011.
[12]. S.Einakian and M. Ghanbari, [2006] "Parallel Implementation of Association Rule in Data Mining", Proceedings of the 38th Southeastern Symposium on System Theory Tennessee Technological University Cookeville, TN, USA, March 2006,pp. 21-26.
[13]. Sharma, Manik, Gurdev Singh, and Harsimran Kaur. [2012] "A study of BNP parallel task scheduling algorithms metric's for distributed database system", International Journal of Distributed and Parallel Systems 3.1 (2012): 157.
[14]. Sharma, Manik, Singh Gurvinder, Singh Rajinder, Singh Gurdev, [2013] "Stochastic Analysis of DSS Queries for a Distributed Database Design", International Journal of Computer Applications 83.5 (2013): 36-42.
[15]. Sharma, Manik, Gurdev Singh, and Rajinder Virk. [2012] "Analysis of Joins and Semi Joins in a Distributed Database Queries", International Journal of Computer Applications 49.16 (2012).
[16]. Tassa T., [2014] "Secure Mining of Association Rules in Horizontally Distributed Databases", IEEE Transactions on Knowledge and Data Engineering, vol. 26 , no. 4,pp.970-983,April2014.
[17]. Parallel Processing, TechTarget, Internet: https://www.techtarget.com/searchdatacenter/definition/parallel-processing.
[18]. Wu Sai, Li Feng, Mehrotra Sharad, Beng Chin Ooi, [2011] "Query optimization for massively parallel data processing", Proceedings of the 2nd ACM Symposium on Cloud Computing. 2011.
[19]. Yoon-Min Nam, Donghyoung Han, Min-Soo Kim, [2019] "A parallel query processing system based on graph-based database partitioning", Information Sciences 480 (2019) 237-260.