GEORGE BROWN PRESS
JOURNALS, BOOKS & PROCEEDINGS

Journal of Computer, Signal, and System Research

Article

Research on Parallel Execution Techniques for Improving the Expandability of Database Systems

Zhongqi Zhu [1,*]

[1] Meta, Infrastructure, Menlo Park, California, 94025, US
* Correspondence: Zhongqi Zhu, Meta, Infrastructure, Menlo Park, California, 94025, US

Abstract: In today's era of widespread big data and cloud computing technology, database systems are under tremendous pressure to handle massive amounts of data and complex queries. When traditional independent database systems encounter high concurrency and large-scale data operations, their performance begins to feel inadequate. Parallel execution technology has been developed as a means to improve database performance and scalability. This technology significantly improves the system's processing capability and response speed by splitting database queries and transaction processing tasks into multiple small tasks for parallel execution. The purpose of this article is to analyze in depth the role of parallel execution technology in enhancing the scalability of database systems, evaluate the current status and challenges of this technology in database applications, and propose improvement strategies for multi-core and multi node architectures. Through in-depth research on the principles and applications of parallel execution technology, the aim is to provide theoretical support and practical suggestions for improving the scalability of database systems.

Keywords: parallel execution; database system; scalability; multi core architecture

1. Introduction

With the explosive expansion of information volume, conventional databases have encountered performance constraints when dealing with large datasets and high-frequency concurrent requests. Parallel execution technology, as a core approach to enhance the scalability of database systems, greatly improves the processing capacity and response speed of the system by subdividing complex tasks into numerous small tasks for parallel execution. How to allocate resources reasonably, maintain load balancing, and ensure data consistency in multi-core processors and distributed computing architectures remains a challenging problem to be solved. This article will delve into the application of parallel execution technology in enhancing the scalability of database systems, analyze the advantages and disadvantages of existing technologies, and provide targeted improvement suggestions, aiming to provide theoretical support and operational guidance for optimizing database performance [1].

2. Overview of Parallel Execution Technology

2.1. Basic Concepts and Principles of Parallel Execution

Parallel execution is the process of breaking down a computing task into several sub tasks that can be executed in parallel, and then utilizing the collaboration of multiple processing units to complete the overall task. In database management systems, data queries and transaction processing are often the most complex and resource consuming, which makes parallel execution technology play a crucial role in these processes. The core idea of parallel execution encompasses task segmentation, rational allocation of resources, even distribution of loads, and integration of final results. Task segmentation refers to

Received: 25 March 2025
Revised: 01 April 2025
Accepted: 18 April 2025
Published: 22 April 2025

breaking down data queries or transactions into multiple smaller tasks that can run independently. Resource allocation involves dispersing these subtasks to different computing nodes or processors to achieve optimal resource utilization. The purpose of load balancing is to prevent individual computing nodes from bearing excessive loads and ensure balanced distribution of tasks among nodes. The integration of results is the consolidation of all parallel executed subtask results to obtain the final query or transaction processing result [2].

2.2. Application Fields of Parallel Execution Technology

Parallel execution technology has been widely applied in many computationally intensive and data intensive industries, especially in the field of database management systems, playing an indispensable role. In terms of database query optimization, using parallel execution technology to optimize query execution plans can effectively improve the speed of processing complex data queries. For example, splitting and executing connection operations through parallel strategies can significantly reduce the time required for queries [3]. Parallelizing the operations of multiple transactions during transaction processing enhances the processing capability of the database system and meets the requirements of high concurrency transactions. When processing massive amounts of data, distributed database systems commonly use parallel execution technology to store data distributed across numerous nodes and accelerate data processing speed through parallel execution. In distributed database systems, the partitioned storage of data and the operation of distributed computing platforms highly rely on parallel execution technology to achieve data exchange and task allocation between nodes, thereby optimizing the overall performance and scalability of the database system. It can be seen that parallel execution technology plays a key role in enhancing database performance and improving system scalability [4].

3. Challenges and Requirements for Scalability of Database Systems

3.1. Analysis of Database Performance Bottlenecks

Faced with the rapid expansion of data scale, traditional database architectures are facing increasing challenges in terms of performance. The first and foremost is the data storage link, where the input and output speed of the disk has become the core factor restricting performance [5]. The dramatic increase in data volume makes it difficult for a single storage device to meet the requirements of high concurrency scenarios, resulting in longer response times for queries. The execution of complex query tasks in databases, such as association, sorting, and summarization, requires a significant amount of computing resources, especially when dealing with massive amounts of data. The sequential execution of these operations can greatly slow down the system's response speed. The efficiency of transaction processing has also become a major obstacle to performance. In an environment where multiple users operate simultaneously, conflicts between transactions and competition for lock resources often lead to an increase in transaction waiting time, thereby reducing the overall processing capacity of the system. With the continuous expansion of database scale, how to efficiently manage and allocate computing resources to prevent system single point failures has become an urgent problem to be solved.

3.2. Key Issues in Scalability Design

When designing the scalability of a database system, it is necessary to address a series of core challenges that play a decisive role in the system's operational efficiency and reliability during scale expansion. Overcoming these challenges is the key to enabling the system to support massive amounts of data, cope with high-intensity concurrent operations, and adapt to changing workloads. Table 1 is the core challenge that database systems must face in the pursuit of scalability.

Table 1. Key Issues in Database Scalability Design.

key issue	describe
Data distribution and partitioning	How to efficiently distribute data to multiple storage nodes and avoid data bottlenecks
load balancing	How to evenly distribute workload among multiple nodes and avoid overloading one node
Transaction management and consistency	How to ensure data consistency and transaction isolation in high concurrency environments
Resource scheduling and dynamic expansion	How to dynamically adjust computing and storage resources based on real-time load requirements

These key issues comprehensively involve multiple levels of database systems in the field of scalability design. The distribution and partitioning of data are key to ensuring storage and retrieval efficiency, load balancing is an important means to maintain smooth system operation, transaction management and consistency are the guarantee of accurate data, and resource scheduling and dynamic expansion are necessary conditions to cope with load changes. Overcoming these challenges is fundamental to building an efficient and flexible database system.

3.3. Impact of Parallel Execution on Database System Scalability

Parallel execution technology greatly enhances the scalability of database systems. By breaking down complex queries or transactions into numerous subunits that can work simultaneously, databases can achieve greater efficiency in multi-core processors and distributed computing architectures, greatly improving processing efficiency. For example, in the data query process, traditional sequential processing may cause query latency, while with the help of parallel technology, databases can synchronize query tasks across multiple processing units, significantly reducing response time. Transaction processing can also achieve efficient concurrent operations through multithreading technology, reducing competition for lock resources and improving the overall processing speed of the system [6]. Parallel technology can also optimize the allocation of computing resources and reduce resource idle. For example, in multi-core and multi node environments, reasonable task allocation and load balancing can ensure the maximum utilization of computing resources. The introduction of parallel execution technology can enhance the processing capability of database systems, shorten response time, and optimize the system's ability to handle large amounts of data and concurrent scenarios, effectively enhancing the scalability of database systems.

3.4. Scalable Resource and Environmental Requirements

In order to ensure good scalability of database systems, in addition to relying on cutting-edge technology and ingenious design ideas, it is also necessary to rely on abundant material resources and appropriate external conditions [7]. The strong support of hardware facilities, software platforms, and system architecture constitutes the core of database scalability. At the hardware level, the performance of the processor, storage space, and network transmission rate are the direct factors determining the scalability potential of the database system. As for the software environment, the construction of database management systems must be compatible with distributed processing, transaction control, and flexible resource allocation in order to effectively handle large-scale data challenges. Table 2 lists the main resources and environmental conditions necessary to achieve database scalability.

Table 2. Resource and Environment Requirements for Scalability of Database Systems.

Resources/Environment	describe
computing resource	Database systems need to support multi-core processors or distributed computing architectures to fully utilize parallel computing to improve performance.

Storage resources	An efficient distributed storage system is the foundation for expanding database storage capabilities, capable of supporting the storage and fast access of massive amounts of data.
network resource	A high bandwidth and low latency network environment can improve the data transmission efficiency between database nodes and reduce communication bottlenecks.
Database management system support	Support distributed transactions, load balancing, dynamic scaling, and other functions to ensure data consistency and system stability during scaling.

The scalability of a database relies on the full coordination of various resources. The processing power comes from computing resources, and performance is enhanced through multi-core and distributed computing architectures. The importance of storage resources lies in ensuring the secure storage and rapid retrieval of large-scale data. Network resources play the role of guardians of data transmission efficiency, especially in distributed architectures. The support of a database management system is fundamental to its efficient operation, including key mechanisms such as transaction processing and load distribution. Only by properly handling these resource and environmental challenges can the scalability of the database be truly achieved.

4. Application of Parallel Execution Technology in Improving Scalability of Database Systems

4.1. Database Query Parallelization Technology

The parallel execution technology using database queries can refine query tasks into numerous subtasks, achieving parallel execution, thereby improving query speed and shortening response time. In the scenario of dealing with massive data and high concurrency requests, parallel query technology greatly enhances the processing capacity and scalability of databases. This technology is mainly divided into two aspects: parallelization of query plans and parallelization of query execution. In query plan parallelization, complex query steps such as concatenation, sorting, and summarization are refined into multiple subtasks and synchronized across different computing units. For example, when performing a connection operation, the data is divided into multiple parts, distributed to different units for processing, and finally the results are summarized to shorten the query time. The parallelization of query execution involves breaking down a single query action into multiple small tasks and synchronously executing them on different units, such as distributed data reading. Data partitioning plays a crucial role in parallel queries, ensuring that data is evenly distributed across multiple nodes to optimize query efficiency and balance load [8].

4.2. Data Partitioning and Distributed Parallel Processing

The scalability optimization of database systems relies on data segmentation and strategies for distributed parallel execution. Data segmentation technology involves dispersing data to numerous nodes according to specific standards, such as hashing or interval segmentation, ensuring that each node only processes a portion of the data. The core of this strategy lies in improving query speed and reducing the pressure on individual nodes, while utilizing parallel operations to enhance data processing speed. Data segmentation techniques are mainly divided into two categories: horizontal segmentation and vertical segmentation. Horizontal segmentation divides data into rows and stores different parts of the data at different nodes. Vertical segmentation allocates data according to columns, mainly for scenarios where only specific columns need to be queried. Through precise data segmentation strategies, databases can effectively allocate storage and computing tasks among multiple nodes, achieving parallel processing of data.

The formula for distributed parallel processing can be expressed as:

$$T_{total} = \sum_{i=1}^{n} \left(\frac{D_i}{P_i} \right) \quad (1)$$

Among them, T_{total} is the total processing time, D_i is the amount of data in each partition, P_i is the number of processing units used on each partition. The increase in nodes enables data to be more evenly distributed and stored, thereby reducing the pressure on individual nodes and improving the processing efficiency of the entire system. A major highlight of distributed processing is its ability to increase the concurrency level of databases by adding nodes as the scale expands, adapting to the challenges of large-scale data and high concurrency processing. With the emergence of problems such as data synchronization and communication networks in distributed systems, how to create efficient distributed query strategies and resource allocation schemes remains a key issue in the field of scientific research.

4.3. Coordination Mechanism between Database Transactions and Parallel Execution

In multi-threaded operation scenarios, transaction synchronization of databases is the core link to ensure data integrity and system reliability. Although multi-threaded operations can enhance the processing power of the system, concurrent operations may cause conflicts, which requires an efficient synchronization strategy to handle. Parallel transaction processing usually adopts two strategies, active synchronization and passive synchronization. Active synchronization allows transactions to occur simultaneously, and conflicts are checked during the transaction commit phase. If there are no conflicts, the commit is completed, and if there are, rollback and retry are performed. This strategy is suitable for use in situations where conflicts occur less frequently and can effectively improve the system's concurrent processing capabilities. Passive synchronization, on the other hand, uses a locking mechanism to ensure that transactions cannot modify the same data during execution. Although this can reduce conflicts, it may also cause lock competition issues, which can affect performance, especially in high load situations. For distributed database systems, the two-stage commit (2PC) protocol is commonly used to maintain transaction consistency and indivisibility. Although it can ensure transaction consistency between nodes, it also comes with a significant performance burden, especially in cases of high network transmission latency. The parallel processing of database transactions requires a balance between consistency and efficiency to ensure smooth transaction processing and accurate data.

4.4. Parallel Execution Optimization under Multi-Core and Multi Node Architectures

With the continuous advancement of technology, multi-core processors and multi node distributed systems have become widely adopted architectural patterns in database systems. How to optimize task allocation, resource management, and ensure data synchronization within such a structural system is the core of improving database parallel processing efficiency. On a multi-core architecture, database systems refine tasks into multiple small task units and distribute these units to different cores for parallel processing, in order to achieve actual parallel computing. In order to maximize the performance of multi-core processors, database systems must have efficient task allocation mechanisms to maintain workload balance among different cores, thereby reducing resource idle rates and improving computational efficiency. In a multi node distributed architecture, the reasonable distribution of data and appropriate allocation of tasks have a decisive impact on the parallel performance of the system. By implementing efficient resource management strategies, reducing communication overhead and resource contention between nodes, database systems can maintain excellent performance even when expanding horizontally.

The optimization under multi-core and multi node architectures can be described by the following formula:

$$T_{\text{parallel}} = \frac{T_{\text{serial}}}{N_{\text{cores}}} \tag{2}$$

Among them, T_{parallel} is the total time for parallel execution, T_{serial} is the time for serial execution, and N_{cores} is the number of processor cores.

With the increase in the number of processor cores, the theoretical parallel execution time should be shortened. However, the actual operational efficiency is constrained by multiple factors such as the rationality of task allocation, load balancing, and communication costs. Optimizing task allocation and resource regulation for multi-core and multi node architectures has become the core of improving parallel efficiency. With the improvement of these advanced technologies, databases can more fully utilize contemporary hardware structures, thereby achieving excellent scalability and performance to cope with the challenges of massive data and high concurrency processing.

5. Conclusion

Parallel execution technology plays a crucial role in enhancing the scalability of database systems. Faced with the constantly increasing data scale and concurrent access requirements, conventional database architectures often encounter performance limitations, making it difficult to efficiently handle massive datasets. By decomposing complex tasks into several small tasks for parallel execution, this technology significantly enhances the processing power and response speed of the database, thereby improving the scalability of the system. Relying on multi-core processing technology and distributed system architecture, databases can fully utilize hardware performance and optimize operational efficiency. With the advancement of processor technology and the maturity of parallel processing theory, database systems are expected to rely more deeply on parallel execution technology to address the challenges of big data processing and real-time decision-making, thereby promoting the continuous progress of database technology.

References

1. R. D. Alessio, A. Giordano, G. Mazzuca, et al., "Tailoring load balancing of cellular automata parallel execution to the case of a two-dimensional partitioned domain," *J. Supercomput.*, vol. 79, no. 8, pp. 9273–9287, 2023, doi: 10.1007/s11227-023-05043-3.
2. W. Liu, L. Lin, J. Zhang, et al., "Multi-core parallel architecture design and experiment for deep learning model training," *Multimedia Tools Appl.*, vol. 81, no. 8, pp. 11587–11604, 2022, doi: 10.1007/s11042-022-12292-6.
3. O. A. M. Khashan, N. M. Khafajah, W. Alomoush, M. Alshinwan, S. S. Atawneh, and M. K. Alsmadi, "Dynamic multimedia encryption using a parallel file system based on multi-core processors," *Cryptography*, vol. 7, no. 1, 2023, Art. no. 12. DOI: 10.3390/cryptography7010012, doi: 10.3390/cryptography7010012.
4. S. Dirim, O. O. Oezener, and H. Soezer, "Prioritization and parallel execution of test cases for certification testing of embedded systems," *Softw. Qual. J.*, vol. 31, no. 2, 2023, doi: 10.1007/s11219-022-09594-1.
5. C. Xia, J. Zhao, and H. F. X. Cui, "HOPE: a heterogeneity-oriented parallel execution engine for inference on mobiles," *High Technol. Lett.*, vol. 28, no. 4, pp. 363–372, 2022, doi: 10.3772/j.issn.1006-6748.2022.04.004.
6. T. Bagies, W. Le, and S. A. Jannesari, "Reducing branch divergence to speed up parallel execution of unit testing on GPUs," *J. Supercomput.*, vol. 79, no. 16, pp. 18340–18374, 2023, doi: 10.1007/s11227-023-05375-0.
7. S. Baheti, P. S. Anjana, S. Peri, et al., "DiPETrans: A framework for distributed parallel execution of transactions of blocks in blockchains," *Concurrency Comput. Pract. Exp.*, vol. 34, no. 10, 2022, doi: 10.1002/cpe.6804.
8. G. Dhanabalan, S. T. Selvi, and M. Mahdal, "Scan time reduction of PLCs by dedicated parallel-execution multiple PID controllers using an FPGA," *Sensors*, vol. 22, no. 12, 2022, doi: 10.3390/s22124584.