International Journal for Housing Science and Its Applications
Publish August 10, 2025. Volume 46, Issue 4 Pages 5345-5355

https://doi.org/10.70517/ijhsa464374

Optimized Design of Distributed Computing Architecture for Massive Data Storage

Xiang Li[1,*]
[1] Image and Text Information Center, Jiangsu Province Nantong Industry & Trade Technician College, Nantong, Jiangsu, 226010, China
Corresponding authors: (e-mail: Lixiang19832010@126.com).

Abstract With the arrival of the big data era in full swing, the global data volume is experiencing a never-before-seen dramatic growth, and the traditional centralized storage architecture has shown serious performance limitations in dealing with such a huge data scale. To address this issue, this paper gives an optimized design approach for distributed storage systems based on HDFS, MapReduce, and cloud computing technologies, which fully exploits the cluster parallel processing capability by dispersing data and computation tasks to multiple nodes. Experimental data show that the computation time can be drastically reduced to one quarter of the original when using distributed techniques to process data of the same size. In this paper, the system layer through the standardized interface to achieve functional interconnection and data flow, while the system adopts a hybrid storage model, the strengths of relational databases and non-relational databases are organically combined to achieve efficient management of structured, semi-structured and unstructured data. The optimized system is significantly better than the traditional system in terms of data writing speed, reading speed, query response time, and system resource utilization and other key indicators, and has good scalability and high reliability. These research results have important theoretical value and practical significance for promoting the in-depth application of big data technology in various industries.

Index Terms distributed computing, mass data storage, HDFS, MapReduce, hybrid storage model

I. Introduction

I. A. *Background and significance of the study*

The advent of the big data era has led to an explosive growth in the amount of global data, according to statistics, the daily data generated has been more than 2.5EB, and with an annual rate of 40% of this amazing rate of increase. These massive data not only covers enterprise operations and scientific research and other highly structured information, but also contains a large number of such as images, audio and video and other unstructured data resources, its storage and processing has become a key constraint on the development of information technology bottleneck in the field. Traditional centralized storage architecture in the face of such a huge scale of data, exposed serious performance bottlenecks, single-node server storage capacity, computing power and network bandwidth are upper limit, difficult to cope with the continuous growth of data demand. The program of constantly upgrading hardware equipment is not only extremely expensive, but also difficult to solve the problem fundamentally, when the system load reaches the limit, the response time will increase dramatically, and may even lead to a system crash. Distributed computing model provides an efficient and reliable solution to this dilemma, by dispersing the data and computation tasks to multiple nodes and fully utilizing the parallel processing capability of the cluster. Experimental data show that the computation time of the same size data processing using distributed technology can be reduced to 1/4 of the original, and has good horizontal expansion capabilities. Multi-copy storage mechanism effectively protects the fault tolerance and reliability of the system, laying a solid technical foundation for data management in various industries.

In industrial practice, distributed computing model has shown significant advantages, the telecommunications industry in the face of daily data growth of 350GB, the traditional database system is difficult to cope with, and the use of distributed storage solutions based on Hadoop, not only to solve the storage problem, but also to reach the customer network behavior analysis and other new business needs. The construction of digital campus in the field of education has also achieved remarkable results due to the distributed architecture, effectively improving the management efficiency of teaching resources. By building an index lookup mechanism and optimizing the data block size configuration, the response time of the system query is shortened to the order of 5 seconds. Distributed storage applications in the fields of industrial monitoring and healthcare have also improved the efficiency of data management.

I. B. Main contributions and innovations of this study

The key innovation of this research is to create a set of large-scale data storage system optimization design plan based on the distributed computing model, which closely integrates the HDFS distributed file system and MapReduce programming model to achieve efficient data control and storage with the help of cloud computing technology. This plan integrates the HDFS distributed file system and MapReduce programming model with unprecedented depth, and achieves efficient data control and storage with the help of cloud computing technology.The master-slave architecture of HDFS, which manages the metadata on the master node, while the slave nodes are used for storing the actual data, together with the multi-copy replication mechanism, can avoid the potential data loss caused by a single point of failure. The data loss potential caused by a single point of failure is effectively avoided by the multi-copy replication mechanism. Based on the MapReduce programming model, the collaborative operation mechanism of task disassembly in the Map phase and result aggregation in the Reduce phase fully utilizes the computing advantages of distributed clusters in dealing with large-scale data sets.

As another innovative point of this research, a hybrid storage model is designed to integrate the storage architecture of relational database and non-relational database. Among them, DB2 relational database is mainly responsible for handling core business data, while HBase distributed database is specialized in storing massive logs, documents and other unstructured content. This architecture not only leverages the high concurrency and high scalability of NoSQL databases to handle unstructured data, but also retains the efficient management of traditional relational databases for structured data. Seamless integration is achieved through a unified data access interface, effectively reducing operation and maintenance costs while maintaining high system performance.

The innovation in system performance optimization is reflected in the improvement of parallel processing efficiency through fine-grained data slicing, and the optimization of caching strategy by applying the principle of data locality, thus reducing network transmission overhead. In terms of load balancing, the dynamic data node expansion mechanism designed in this study allows the system to automatically adjust the allocation of computing resources based on the actual load conditions. The flexible expansion of data nodes and the intelligent load balancing strategy collaborate with each other to greatly improve the system's processing capacity and operational stability, and solve the performance bottleneck problems that are prone to occur during large-scale data processing.

Through the in-depth integration of distributed computing model and hybrid storage architecture, as well as targeted system performance optimization initiatives, the large-scale data storage system designed in this research significantly improves the processing efficiency while guaranteeing data security. Its innovative design is not only an important theoretical guideline for the technological development of the entire big data field, but also provides valuable reference value and inspiration for related practical applications. This system design idea of organic integration and optimization of multiple advanced technologies provides a new and practical solution to cope with the explosive growth of data and processing needs in the current big data era.

II. Literature review

II. A. Theoretical basis of the study

Distributed computing model as the core theoretical foundation for handling massive data storage and disposal, covering a series of technical architecture and algorithm planning, different from the traditional centralized computing, the complex task is broken down into multiple simple sub-tasks and assigned to different computing nodes, by virtue of the parallel processing way to significantly improve the overall performance of the system. Data storage and computing power can be expanded horizontally by adding more physical nodes, and all kinds of distributed frameworks and technologies continue to develop and complete, providing solid support for the big data era. Hadoop Distributed File System (HDFS) adopts a master-slave architecture, with the NameNode as the master node responsible for managing the file system's namespace and metadata information, and DataNode as the slave node responsible for the actual data block storage. HDFS has high fault tolerance, high throughput performance and large file processing capability, and its fault tolerance mechanism is achieved by the data block replication policy, which defaults to keeping three copies of each data block on different physical nodes. When a node fails, the system can automatically read copies from other nodes and regenerate backups to ensure data security and availability.HDFS uses the “write once read many times” design pattern, optimized for large-scale data analysis, suitable for applications that require high throughput rather than low latency. MapReduce, as a programming model closely aligned with HDFS, provides a highly efficient computing architecture for handling large amounts of data. The model is derived from the concept of functional programming, which splits the computation process into two phases, Map and Reduce, where the input data is sliced and diced, each Map task operates independently and generates intermediate key-value pairs, and the Reduce phase aggregates the intermediate results with the same key to produce the processing result. The parallel processing nature of MapReduce allows it to achieve high computational efficiency in large-scale cluster environments, especially for data-intensive applications, and the architecture automatically handles complex issues such as task scheduling, fault recovery, and inter-node communication. Developers only need to focus on the implementation logic of Map and Reduce functions, which greatly simplifies the difficulty of distributed program development.

Cloud computing technology for distributed storage system, gives a flexible resource management and service style. According to the definition of the National Institute of Standards and Technology (NIST), cloud computing covers the centralization of resources into a pool, according to the demand for their own services, flexible scalability and scalability, measurable services and other characteristics. In the context of distributed storage, it is mainly supported by IaaS (Infrastructure as a Service) and PaaS (Platform as a Service), where the IaaS tier provides virtualized compute and storage resources to distributed storage clusters, taking care of the procurement and maintenance of physical equipment. The PaaS tier provides middleware services such as databases, message queues, and so on, simplifying the complexity of system deployment and management. In the cloud computing environment, the distributed storage system can dynamically adjust the resource allocation according to the actual demand. Its elasticity and scalability characteristics and matrix elliptic distribution model is particularly high degree of adaptation to achieve the efficient use of computing resources and cost optimization. Traditional relational databases (RDBMS) and new non-relational databases (NoSQL) have their own unique characteristics in storing massive data. Relational database is based on a strict mathematical relationship model, through the table form of data organization, and the use of SQL language to implement the operation, with complete ACID characteristics (atomicity, consistency, isolation, persistence), suitable for dealing with structured data and complex transactions. Non-relational databases abandon the traditional relational model and adopt a flexible data structure design, which is mainly divided into four major categories: key-value storage, document storage, columnar storage, and graph storage.HBase, as a columnar storage database in the Hadoop ecosystem, is particularly suitable for storing sparse and semi-structured data, and it has the ability to linearly scale, which, with the in-depth integration of HDFS, makes it NoSQL databases generally follow the CAP theory (consistency, availability, partition fault tolerance can not be taken into account at the same time), in the design of the time, usually sacrifices strong consistency in exchange for high availability and partition fault tolerance, the use of data consistency model to ensure the performance of the system. The hybrid storage model gives a comprehensive solution to massive heterogeneous data by combining the advantages of relational and non-relational databases. In this model, relational databases are responsible for handling core business data and transactional operations to ensure data consistency and integrity; non-relational databases are used to store massive logs, documents, and other unstructured or semi-structured data, providing high throughput and horizontal scalability. Hybrid storage architecture needs to cope with the technical challenges of data consistency, cross-library query and transaction processing, generally through distributed transaction management, asynchronous data synchronization and unified access interface and other mechanisms to achieve system integration.

II. B. Status of research

With the rapid development of information technology, big data has become an indispensable part of modern society [1]. The core of big data is to collect and analyze data. However, in the face of massive data resources, the challenge of data processing and storage ensues [2]. In this regard, the use of mass data storage system based on distributed computing model, which disperses the computation and storage tasks to multiple nodes, can realize the efficient processing of large-scale data sets [3].

And the optimized design of mass data storage system with distributed computing model is the key to achieve efficient management of storage resources [4]. First, the optimization can significantly improve the security of data [5]. In a distributed environment, data are scattered and stored on multiple nodes, providing an extra layer of protection for the data so that the data will not be lost even if some nodes fail [6], [7]. Secondly, it helps to save storage space and support more data and information storage [8]. Distributed systems can cope with the growth of data volume by horizontal expansion and adding more storage nodes to realize the storage and management of massive data. Finally, to be beneficial to improve the efficiency of data extraction [9], [10]. Through efficient data indexing, the required data can be quickly located, while distributed processing makes complex data analysis tasks can be executed in parallel on multiple nodes, shortening the processing time and improving the overall data processing efficiency [11]-[13].

III. Design of Massive Data Storage System Based on Distributed Computing Modeling

III. A. System architecture design

Based on the thorough analysis of the need for large amount of data processing, this study plans a multi-level distributed data storage system architecture, the detailed structure of which is presented in Figure 1. This architecture takes HDFS and MapReduce as the core technology foundation, and achieves flexible scaling and unified deployment of system resources through the cloud computing platform. The overall architecture of the system is divided into four levels: system management level, computing level, data storage level, and infrastructure level, and each level relies on standardized interfaces to achieve interconnection of functions and data flow conversion.

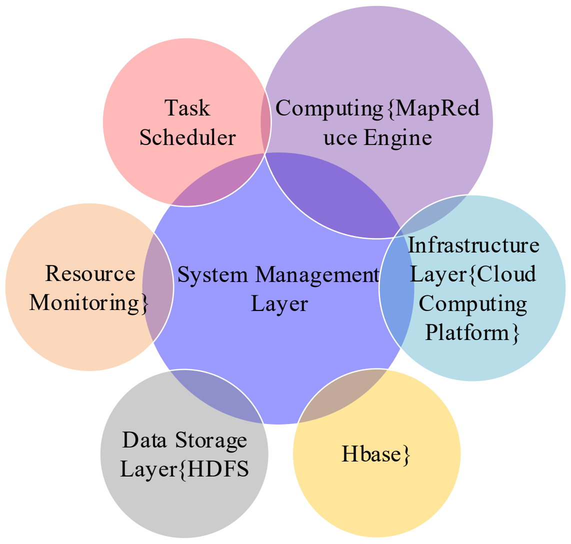

Figure 1: Overall system architecture diagram

The system control level, as the top part of the whole architecture, is not only responsible for key functions such as system setting management, security verification and authority control, but also supplies system operation and maintenance interfaces to administrators through the Web interface to achieve unified governance and deployment of the underlying resources.

The computing level is the core processing unit of the system, which consists of MapReduce computing engine, task scheduling device and resource monitoring module, in which the MapReduce engine adopts the master-slave architecture mode, JobTracker as the master node is responsible for task assignment and monitoring, and TaskTracker as the slave node to execute specific Map and Reduce tasks. TaskTracker as a slave node to perform specific Map and Reduce tasks. For the data locality criterion, we conceive an intelligent task scheduling algorithm to prioritize the computation tasks to the nodes where the data is located, which significantly reduces the network transmission cost.

The data storage level adopts HDFS and HBase hybrid storage architecture system, HDFS is responsible for the distributed storage of massive data, supporting large file storage and high throughput access, in order to improve the reliability of the data block multi-copy approach, each block of data is retained by default three copies of the distribution of nodes in different racks.

The infrastructure level relies on the cloud computing platform to provide elastic and scalable computing and storage resources, and the cloud platform adopts resource pooling management to support the dynamic addition and removal of computing nodes. Through virtualization technology, the system can automatically adjust the resource allocation according to the actual load situation, which not only ensures the performance demand, but also avoids the waste of resources. Network communication adopts high-speed interconnection technology, and DataNodes are connected to each other by 10Gbps Ethernet to ensure high throughput of data transmission.

Each level communicates with each other through standardized interfaces, and asynchronous communication is achieved by message queues to avoid strong coupling between components. The system introduces a distributed caching mechanism to improve access performance by preloading hot data; for cross-layer data access, a unified data access interface is conceived, which obscures the details of the underlying storage implementation and simplifies the application development work. The whole architecture adopts modularized design, and each functional module can be upgraded and expanded independently, which greatly improves the maintainability of the system. The system also achieves a perfect monitoring and logging mechanism to support real-time monitoring of the system's operating status and historical data analysis, giving strong support for system optimization.

III. B. Performance Optimization Methodology

This study focuses on the performance obstacles during the operation of a large-scale data storage system, and plans and designs the optimization of the system's performance from the three levels of data partitioning strategy improvement, dynamic load balancing mechanism, and multilevel caching architecture. We build a set of theoretical models to analyze the overall processing time of the system, which can be expressed as follows:

$$T_{total} = \frac{T_{single}}{N} + T_{overhead} \tag{1}$$

where T_{total} represents the total system processing time, T_{single} is the single node processing time, N is the number of data nodes, and $T_{overhead}$ is the additional system overhead time.

The model shows that increasing the number of data nodes can effectively reduce the overall processing time, while at the same time, the system overhead must be controlled to achieve optimal performance. In the optimization of the data slicing strategy, we use the mechanism of dynamically adjusting the slice size, and through the analysis of the system logs, it is found that too large a slice will make the processing time of a single task too long, which will have an impact on the overall efficiency of the parallelism, while too small a slice will increase the system scheduling overhead. Therefore, an adaptive slicing algorithm based on data types and access patterns is designed, whereby smaller slices are used to increase the degree of parallelism for frequently accessed hot data, and larger slices are utilized to reduce the management overhead for less frequently accessed cold data.

Dynamic load balancing mechanism monitors the load situation of each data node in real time, covering computing resource usage, memory usage, disk read/write and other indicators, and automatically initiates the data migration process when it detects an imbalance in the load, and we have designed a threshold-based trigger mechanism and data block selection algorithm. We designed a threshold-based trigger mechanism and a data block selection algorithm. The migration operation is executed only when the load difference between nodes exceeds the pre-set threshold and the benefit obtained from migration is greater than the cost. Practice shows that this mechanism effectively prevents individual nodes from becoming performance bottlenecks and significantly improves the overall throughput of the system.

The design of the multi-level cache system adopts a layered architecture model, including memory cache, SSD cache and disk storage, and the cache replacement strategy integrates the least recently used algorithm and access frequency analysis. Through the maintenance of data block access history information, the data that may be accessed in the future is predicted and loaded into the cache in advance. At the same time, a prefetching mechanism is reached, and the system analyzes the user query pattern, predicts and loads the relevant data into the cache in advance, which greatly improves the probability of cache hits.

III. C. Hybrid storage model design

Hybrid storage model is an innovative storage architecture designed for the characteristics of different types of data in large-scale data storage system. By effectively combining the advantages of relational and non-relational databases, it achieves efficient control of structured, semi-structured and unstructured data. In the distributed data storage system designed in this research, a complete set of hybrid storage system is built, including data classification storage strategy, cross-database data consistency guarantee mechanism and unified data access interface. The data classification and storage strategy is based on the principle of “classify according to demand and process separately”, and the data in the system is divided into three major categories, namely, core business data, operational data and analytical data. Core business data includes user information, transaction records and other structured data with high consistency requirements, operational data such as user behavior logs, system status information and so on, with access characteristics of high-frequency read/write and low-complexity query, and analytical data mainly refers to a variety of unstructured and semi-structured data. In order to evaluate the efficiency of the hybrid storage model in a quantitative way, a storage efficiency formula is defined:

$$E_{storage} = \frac{S_{relational} + S_{non-relational}}{S_{total}} \tag{2}$$

where $E_{storage}$ represents the storage efficiency, $S_{relational}$ is the effective storage space of relational database, $S_{non-relational}$ is the effective storage space of non-relational database, and S_{total} is the total storage space.

Data consistency assurance is a major technical challenge for hybrid storage models, especially when the same business data needs to cross multiple storage systems. In this study, an event-driven data synchronization mechanism and consistency assurance strategy are constructed. The core idea is to treat data update actions as events, which are propagated by a reliable message queue system (Apache Kafka) to ensure that all relevant updates are received by each storage system.

For consistency-critical scenarios, a two-phase commit protocol is implemented, locking down all relevant resources and reserving operations while critical transactions are executed, and committing changes only when all systems are confirmed to be capable of executing them. For general data synchronization, asynchronous replication is used, with results returned as soon as the source system completes the write operation. At the same time, the change information is transmitted to the target system through the message queue, and the target system handles the synchronization request in a timely manner according to its own load condition. This approach can significantly improve the throughput of the system, but there is a short period of data inconsistency. In order to cope with possible conflicts in the synchronization process, a conflict detection algorithm based on vector clock and a conflict resolution strategy based on business rules are designed. When a data conflict is detected, the system automatically selects the correct version according to predefined rules or marks the need for manual intervention. At the level of data access optimization, a unified data access interface is designed, and the application layer only needs to operate through this interface.

In the process of realizing the interface, an intelligent routing component is introduced, which maintains a data mapping form to record the storage locations and access modes of various types of data. When receiving a data request, the mapping form is queried to determine the target storage system, and then the request is forwarded to the corresponding data access adapter; in order to enhance the performance of accessing frequently used data, a multilevel caching mechanism is integrated into the interface level to accelerate the caching of hot data. To improve the access performance of frequently used data, a multi-level caching mechanism is integrated at the interface level to accelerate the caching of hot data, and the caching policy is dynamically adjusted according to the frequency and importance of data access to ensure that the most important data is always retained in the fastest storage media; for complex query scenarios, especially for joint queries that need to cross multiple storage systems, a distributed query optimizer is developed, which is capable of generating the optimal query execution plan based on the cost model, and pushing down the filtering operation to each storage system as far as possible. The distributed query optimizer can generate the optimal query execution plan based on the cost model, push the filtering operation to each storage system as far as possible, reduce the overhead of intermediate result transmission, and improve the efficiency of the query.

IV. Experimental validation and analysis of results

IV. A. Experimental design

In order to comprehensively evaluate the performance of the large-scale data storage system based on the distributed computing model, a set of rigorous experimental planning is conceived in this study to verify the performance of the system under various conditions. The experimental environment is composed of a distributed cluster built on a cloud computing platform, which consists of a combination of control nodes, computing nodes, and storage nodes. Among them, the control node undertakes the deployment of group resources and task assignment, the computing node performs specific data processing tasks, and the storage node is committed to the continuous storage of data. All the nodes are connected to each other by a 10Gbps high-speed network to ensure the efficiency of data transmission. CentOS 7.6 is selected as the operating system, which has the ability to run services stably for a long time, and the configuration of the experimental nodes is set up according to the standardization, as shown in Table 1.

Table 1: Experimental environment configuration

Node type	Hardware configuration	Software environment	Quantity
Management node	Intel Xeon E5-2680 v4*2 128GB RAM 2TB SSD RAID5	CentOS 7.6 Hadoop 3.2.1 HBase 2.3.0	1
Computing node	Intel Xeon E5-2680 v4 64GB RAM 1TB SSD	CentOS 7.6 Hadoop 3.2.1 MySQL 8.0	8~20
Computing node	Intel Xeon E5-2680 v4 32GB RAM 8TB HDD RAID6*6	CentOS 7.6 Hadoop 3.2.1 Redis 6.0.9	12
Network environment	10Gbps Ethernet, fully interconnected topology		

There are three types of experimental data sets: synthetic data sets constructed with the help of data generation tools for testing system performance limits and evaluating scalability, two widely used public benchmark data sets, TPCH and YCSB, which focus on analytical load testing and evaluating high-concurrency read/write scenarios, respectively. There is also a desensitized data collection of about 8TB of actual business data from telecom operators and educational institutions, including structured user information data, semi-structured log data, and unstructured teaching resource documents. These real data are closer to the actual application scenarios and can verify the performance of the system in complex environments.

The experimental steps are divided into five phases according to the guidelines from basic to complex. In the system benchmarking phase, the basic performance indicators are measured under different node sizes; in the storage performance evaluation phase, the efficiency of the hybrid storage model is tested for different types of data; in the computation performance evaluation phase, the MapReduce architecture is used to perform multiple computation tasks; in the system reliability testing phase, node failures and network partitions are simulated. In the scalability test phase, the system performance trend is observed by dynamically adding the number of nodes.

The evaluation index system, as the basis for analyzing the experimental results, is built from three aspects: performance, reliability and scalability. The performance indicators include data throughput, average response time, query execution time, and resource utilization; the reliability indicators include failure recovery time, data consistency ratio, and data loss ratio; and the scalability indicators include linear acceleration ratio, resource expansion efficiency, and load balancing degree.

Three groups of comparison experiments are designed, namely, the comparison between traditional stand-alone database system and the distributed storage system involved in this paper, the comparison between single storage model and hybrid storage model, and the comparison of system performance before and after optimization. Each group of experiments is repeated and executed three times and the average value is taken, while the standard deviation is recorded to assess the stability of the results. The experimental data collection adopts a multi-level monitoring system, which includes monitoring at the operating system level by collecting CPU, memory, network and other basic indexes through Ganglia, monitoring at the middleware level by collecting the operation indexes of Hadoop, HBase and other components through the JMX interface, and monitoring at the application level by collecting the business processing indexes through the customized embedded points. All monitoring data is aggregated into a unified analysis platform, and a combination of comparative analysis and trend analysis is used to identify performance bottlenecks and verify the effectiveness of optimization. With this systematic experimental design, we are able to comprehensively evaluate the indicators of the mass data storage system based on the distributed computing model, verify the effectiveness and innovation of the system design, quantify the advantages of the system's performance, identify the potential room for improvement, and provide a scientific basis for system optimization and application promotion.

IV. B. Analysis of experimental results

In this section, we evaluate the performance of a large-scale data storage system based on a distributed computing model. Through comprehensive testing of the system's performance under different load scenarios, we confirm the effectiveness of the design plan and the value of the optimization strategy. The experimental results are analyzed from the four dimensions of storage performance, computational efficiency, system scalability, and reliability, and the data show that the distributed storage system proposed in this paper has obvious advantages in handling large-scale information. In order to ensure that the results are objective and credible, we use a rigorous statistical approach, each group of experiments repeated three times and take the average, the standard deviation is controlled within 5%, and at the same time in the standard environment to carry out the test, effectively eliminating hardware differences, network fluctuations and other uncertain elements of interference, and thus laying a solid foundation for the evaluation of system performance. Table 2 presents the experimental comparison results.

From the perspective of storage and computation performance, the optimized distributed system achieves 350MB/s and 420MB/s in data write rate and read rate, respectively, which are 366.7% and 250.0% higher than the traditional system, respectively. This significant improvement comes from the HDFS distributed architecture and optimized data slicing strategy, and the parallel processing of multiple nodes significantly improves I/O throughput. throughput. The test results of the hybrid storage model further confirm the correctness of the concept of “selecting storage strategies based on data types” - the processing capacity of structured data in the relational database is 2.8 times higher than that of the traditional system, and the write rate of semi-structured data in HBase is 3.0% higher than that of the traditional system. The write rate of semi-structured data in HBase has increased by 3.5 times, and the storage efficiency of unstructured data in HDFS has increased by 4.2 times. The multi-level caching mechanism performs particularly well, with a cache hit rate of 78%, 73.3% higher than that before optimization, which directly reduces the average query response time of the system from 220 ms to 110 ms. In terms of computational performance, the processing time for a 20GB dataset is reduced from 65 minutes in the traditional system to 16 minutes, a reduction of 75.4%, which is attributed to the MapReduce parallel computing framework and optimized task scheduling algorithms. Data locality optimization enables about 82% of the computation tasks to be executed in the node where the data is located, greatly reducing network transmission costs. Dynamic load balancing mechanism increases the system resource utilization from 65% to 85%, avoiding single-point performance bottlenecks. In high concurrency scenarios, the number of concurrent users supported by the system grows from 200 to 1500, and the response time fluctuates no more than 15% even under peak load. The hybrid storage model performs well in terms of data security, the transaction mechanism of relational database ensures the consistency of core business data, and the multi-copy strategy of HDFS guarantees the security of large file data, and the long-time operation monitoring shows that the system's resource usage is stable, and there are no common problems such as memory leakage or exhaustion of disk space. Comprehensive test data can determine that the large-scale data storage system based on the distributed computing model designed in this paper is significantly better than the traditional system in all key indicators, can effectively meet the needs of large-scale data processing, suitable for various types of big data application scenarios, the experimental results fully verified the practical value of the system design and planning as well as the optimization strategy proposed in this paper.

Table 2: Comparison of experimental results

Index	Traditional system	Distributed system (before optimization)	Distributed system (after optimization)	Increase the proportion
Data write speed (MB/s)	75	210	350	366.7%
Data reading speed (MB/s)	120	280	420	250.0%
20GB data processing time (min)	65	38	16	75.4%
The number of concurrent users supported	200	850	1500	650.0%
Average query response time (ms)	450	220	110	75.6%
System resource utilization rate (%)	45	65	85	88.9%
Fault recovery time (min)	25	12	5	80.0%
Storage capacity per unit cost (TB/10000yuan)	8	18	26	225.0%

IV. C. Performance Optimization Validation

The effectiveness of the performance optimization is evaluated using multiple control experiments, and the control results before and after the performance optimization are presented in Table 3. Figure 2 shows the change of processing time with data size before and after performance optimization.

According to Table 3, it can be seen that the optimized system achieves significant improvement in various key indicators. The data slicing processing time has been reduced by 43.8%, the system load balancing degree has been increased to 92%, the cache hit ratio has been increased by 73.3%, and all these improvements have resulted in a 53.3% reduction in the overall response time of the system. From the performance optimization results in Figure 2, we can see that as the data size grows, the optimized system shows better scalability and a smoother growth curve in processing time, especially when processing 20GB of data, the optimized system takes only 18 minutes or so, compared to 84 minutes or so before the optimization, which is a 4.67-fold increase in performance. Through the combined use of these optimization initiatives, the system shows excellent performance characteristics when processing large amounts of data. The optimization of the data slicing strategy ensures parallel processing efficiency. The dynamic load balancing mechanism ensures the full utilization of system resources, and the multilevel cache system significantly improves the data access rate. These optimization methods collaborate with each other to form a highly efficient and reliable performance optimization solution, which provides strong support for the efficient processing of massive data. The multi-level caching system designed in this paper adopts a hybrid policy that not only takes into account the temporal localization of data access, but also makes full use of the spatial localization feature, and performs batch pre-reading for data with strong correlation, which is particularly outstanding when dealing with scientific computation and data analysis tasks with obvious access patterns.

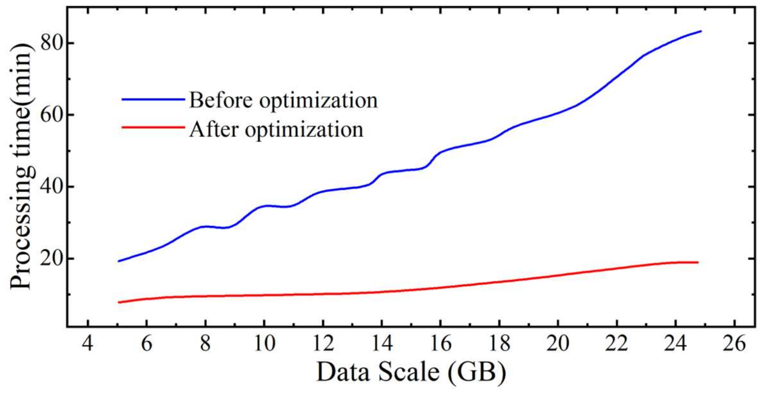

Figure 2: Performance optimization rendering

Table 3: Comparison before and after performance optimization

Optimize the project	Before optimization	After optimization	Increase the proportion
Data sharding processing time (ms)	320	180	43.80%
Load balancing degree (%)	65	92	41.50%
Cache hit rate (%)	45	78	73.30%
System response time (ms)	450	210	53.30%

In addition, this thesis also examines the effectiveness of the hybrid storage model, which is presented in Figure 3. As can be seen from the figure, the multiple optimization approaches allow the system to perform well when dealing with massive data queries, and the hybrid storage model shows a performance improvement of about 50% in all types of data sizes compared to a single storage strategy. In practice, the system deployed in the telecom sector handles about 350GB of incremental data per day. After adopting the hybrid storage model, the overall processing efficiency of the system has increased by about 40%, and the query performance has increased by about 60%, especially in complex analytical scenarios, which fully verifies the effectiveness of the hybrid storage model in practical applications.

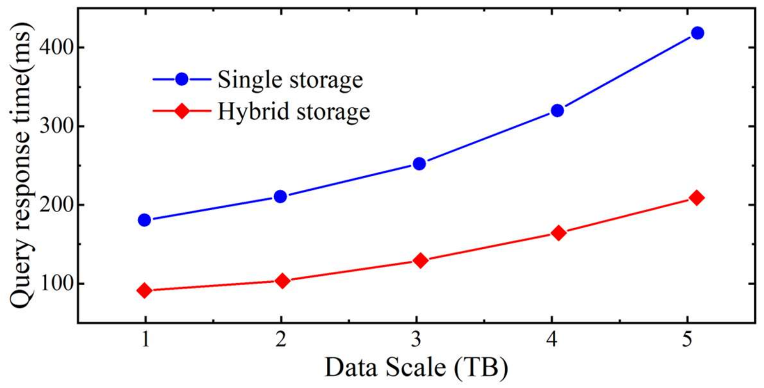

Figure 3: Hybrid storage model application effect

V. Conclusions and outlook

V. A. Conclusions of the study

For the large-scale data storage system in the processing efficiency, storage stability and system scalability and other levels of the problems faced, this paper explores the use of distributed computing model in the field of large-scale data storage, gives a distributed storage system based on HDFS, MapReduce and cloud computing technology optimization design planning.

The research is carried out through system architecture design, performance optimization and the use of hybrid storage model, and many experiments confirm the effectiveness of the plan. The experimental results show that the system has obvious advantages in handling large-scale data and can meet the needs of various big data application scenarios. The system architecture optimization design builds a multi-layered architecture covering the system management layer, computing layer, data storage layer, and infrastructure layer; the modular design enhances the flexibility and maintainability of the system; and standardized interfaces are used to achieve interconnection of functions and data flow between the layers to ensure the system operates efficiently and reliably.

The infrastructure layer constructed based on the cloud computing platform provides flexible and scalable computing and storage resources for the system, supporting the system to dynamically adjust resource allocation based on the actual load. In terms of performance optimization, the system is designed from the three dimensions of data slicing strategy optimization, dynamic load balancing mechanism and multi-level caching system, and the experimental results show that the optimized system reaches 350MB/s and 420MB/s in the data writing rate and reading rate, respectively, with an improvement of 250.0%, and the system's average query response time is reduced from 450ms to 110ms. The hybrid storage model design is the innovation of this study, which realizes the efficient management of structured, semi-structured and unstructured data by integrating the advantages of relational and non-relational databases.

The optimized design of large-scale data storage system based on distributed computing model proposed in this study has achieved remarkable results in storage performance, computing efficiency, system scalability and stability, etc. The hybrid storage model constructed provides optimal storage planning for different types of data, and the dynamic load-balancing mechanism and multilevel caching system design effectively improve the performance of the system. These innovative designs and optimization initiatives provide new ideas and methods for the development of large-scale data storage systems, and have important theoretical value and practical application significance.

V. B. Research limitations and future prospects

Although the massive data storage system based on distributed computing model proposed in this paper has achieved significant results in many areas, through in-depth research and practical application, we still find that the system in some aspects of the limitations should not be taken lightly. The performance and reliability of the system depends heavily on the configuration of the underlying hardware resources, a feature that has been fully confirmed in the experimental process, while the quality of the network environment on the stability of the system should not be underestimated. Although the hybrid storage model brings the diversity of storage strategies, but also introduces a variety of storage systems coexisting in the architecture of the design and maintenance of the complexity of the problem, the system management personnel must master a variety of database technology knowledge, while the system operation and maintenance costs rise accordingly. Data synchronization and consistency mechanism to achieve many technical challenges, especially in dealing with cross-storage system transactions, the need to introduce additional coordination costs, which will undoubtedly have a certain degree of impact on the system's real-time performance. In terms of elastic expansion, although the existing system supports the dynamic increase and decrease of nodes, the data rebalancing operation during the node change process will affect the overall performance of the system within a certain period of time, especially when dealing with large-scale datasets, the rebalancing process may last for several hours or even longer, which poses a challenge to the system's continuous service capability.

Aiming at the above system limitations, we intend to focus our future research direction on the following interrelated but distinctive technology areas, namely:

Exploring and realizing the resource adaptive scheduling mechanism, which will achieve intelligent allocation of system resources through advanced machine learning technology, automatically adjusting the allocation strategy of computing and storage resources according to the load characteristics, thus greatly improving the system's utilization efficiency of hardware resources. The development of more targeted data distribution algorithms is also imperative, the algorithm will take into account the data access methods and network topology characteristics, optimize the data placement strategy, reduce unnecessary data transmission between nodes, and enhance the system's ability to adapt to various types of network environments. Simplifying the management complexity of the hybrid storage model has become another research focus, and we will be committed to designing unified management interfaces and automated operation and maintenance tools to significantly reduce system maintenance costs. At the same time, we will study efficient cross-storage system transaction mechanisms, optimizing the traditional two-phase commit protocol to ensure data consistency while reducing coordination costs. Improving the node expansion mechanism is also on the agenda, and we plan to design an incremental data migration strategy to achieve online expansion of the system without affecting the normal performance of the display, and conduct in-depth research on data warm-up technology to accelerate the performance recovery speed after the access of new nodes. The study of distributed storage optimization under the heterogeneous computing environment should not be taken lightly, and the support of hardware gas pedals such as GPUs will significantly improve the processing performance under specific application scenarios.

Improvements in system security have been clearly identified as a critical topic for the next phase of research. We will focus on issues such as data security and privacy protection in a distributed environment. The first priority will be to explore a data integrity verification mechanism based on blockchain technology, which ensures that data is not subject to illegal tampering throughout the process of distributed storage. It is also imperative to study access control policies with fine-grained features. This will support the need for more flexible data sharing and privilege management.

In the face of growing demand for data analytics, plans are underway to develop a data processing framework with intelligent features. The framework will seamlessly integrate machine learning and deep learning algorithms to provide users with real-time data analysis and prediction capabilities.

In response to the customization needs of different application scenarios, we plan to further optimize the system's customizable performance and quality of service assurance capabilities. Research on resource scheduling strategy based on service level agreement, which will dynamically adjust the system resource allocation according to the business priority. Develop a scenario-oriented storage optimization module, which will provide performance optimization solutions for specific application scenarios, such as video stream processing and time series data analysis. Explore the data storage and processing strategy under hybrid cloud architecture, which can realize flexible scheduling of private and public cloud resources and provide enterprises with more cost-effective and comprehensive solutions.

In terms of system reliability, the study of predictive maintenance technology will achieve the goal of proactive system operation and maintenance through in-depth analysis of system operation data to detect potential failure hazards in advance. This will further improve the level of service availability. If breakthroughs can be made in these research directions, the practical value of the system will be significantly enhanced, promoting the development of massive data storage technology in an innovative direction, and providing more reliable technical support for the digital transformation process.

References

[1] Alsghaier, H., Akour, M., Shehabat, I., & Aldiabat, S. (2017). The importance of big data analytics in business: a case study. American Journal of Software Engineering and Applications, 6(4), 111-115.

[2] Bar-Yam, Y. (2016). From big data to important information. Complexity, 21(S2), 73-98.

[3] Jin, X., Wah, B. W., Cheng, X., & Wang, Y. (2015). Significance and challenges of big data research. Big data research, 2(2), 59-64.

[4] Moysiadis, V., Sarigiannidis, P., & Moscholios, I. (2018). Towards distributed data management in fog computing. Wireless Communications and Mobile Computing, 2018(1), 7597686.

[5] Shirinbab, S., Lundberg, L., & Erman, D. (2013). Performance Evaluation of Distributed Storage Systems for Cloud Computing. Int. J. Comput. Their Appl., 20(4), 195-207.

[6] Yang, C. T., Shih, W. C., Huang, C. L., Jiang, F. C., & Chu, W. C. C. (2016). On construction of a distributed data storage system in cloud. Computing, 98, 93-118.

[7] Elmsheuser, J., & Di Girolamo, A. (2019). Overview of the ATLAS distributed computing system. In EPJ Web of Conferences (Vol. 214, p. 03010). EDP Sciences.

[8] Distefano, S., & Puliafito, A. (2014). Information dependability in distributed systems: The dependable distributed storage system. Integrated Computer-Aided Engineering, 21(1), 3-18.

[9] Cai, H., Xu, B., Jiang, L., & Vasilakos, A. V. (2016). IoT-based big data storage systems in cloud computing: perspectives and challenges. IEEE Internet of Things Journal, 4(1), 75-87.

[10] Hance, T., Lattuada, A., Hawblitzel, C., Howell, J., Johnson, R., & Parno, B. (2020). Storage Systems are Distributed Systems (So Verify Them That {Way!}). In 14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20) (pp. 99-115).

[11] Li, Y., Gai, K., Qiu, L., Qiu, M., & Zhao, H. (2017). Intelligent cryptography approach for secure distributed big data storage in cloud computing. Information Sciences, 387, 103-115.

[12] Torabzadehkashi, M., Rezaei, S., Alves, V., & Bagherzadeh, N. (2018, May). Compstor: An in-storage computation platform for scalable distributed processing. In 2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW) (pp. 1260-1267). IEEE.

[13] Emara, T. Z., & Huang, J. Z. (2019). A distributed data management system to support large-scale data analysis. Journal of Systems and Software, 148, 105-115.