Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 3839800, 14 pages
https://doi.org/10.1155/2021/3839800

Research Article

Block Storage Optimization and Parallel Data Processing and Analysis of Product Big Data Based on the Hadoop Platform

Yajun Wang, Shengming Cheng, Xinchen Zhang, Junyu Leng, and Jun Liu

School of Mechanical Engineering and Automation, Dalian Polytechnic University, Dalian 116034, China

Correspondence should be addressed to Yajun Wang; wangyj@dlpu.edu.cn

Received 27 October 2021; Accepted 10 December 2021; Published 31 December 2021

Academic Editor: Francesc Pozo

The traditional distributed database storage architecture has the problems of low efficiency and storage capacity in managing data resources of seafood products. We reviewed various storage and retrieval technologies for the big data resources. A block storage layout optimization method based on the Hadoop platform and a parallel data processing and analysis method based on the MapReduce model are proposed. A multireplica consistent hashing algorithm based on data correlation and spatial and temporal properties is used in the parallel data processing and analysis method. The data distribution strategy and block size adjustment are studied based on the Hadoop platform. A multidata source parallel join query algorithm and a multi-channel data fusion feature extraction algorithm based on data-optimized storage are designed for the big data resources of seafood products according to the MapReduce parallel frame work. Practical verification shows that the storage optimization and data-retrieval methods provide supports for constructing a big data resource-management platform for seafood products and realize efficient organization and management of the big data resources of seafood products. The execution time of multidata source parallel retrieval is only 32% of the time of the standard Hadoop scheme, and the execution time of the multichannel data fusion feature extraction algorithm is only 35% of the time of the standard Hadoop scheme.

1. Introduction

Owing to the rapid expansion of seafood enterprises, condition-monitoring technologies for mariculture and seafood production and circulation have increased. A large number of intelligent monitoring devices are currently available, and data collected from these devices continue to grow exponentially. For seafood products, the big data resources include the geographic information of mariculture, weather, site temperature, humidity, monitoring videos and images, experimental documents, breeding and processing data, production, and circulation data. Big data resources are large and variable and have scattered distributions [1–3]. A distributed organization and management method is a necessary tool for the management of the big data resources of seafood products. However, the efficiency of traditional distributed database frameworks for the storage and management of data resources is low. Moreover, the storage capacity of systems is limited by the storage capacity of database management systems. The support and release capacity of traditional distributed database frameworks for managing data resources is relatively weak [4–6].

At present, data acquisition and processing systems mostly use traditional relational databases to store data. For example, SQL Server uses single-thread or multithread technology to process data. The performance requirements of data-processing servers are relatively high. Although it can meet the daily management work, it is difficult to meet the data scale and resolution ratio demand of modeling analysis and advanced applications. If the real-time database system is adopted, although it can meet the requirement of data acquisition resolution ratio, with the increase of data volume, the load rate of the server will be relatively high, while the cost of a high-performance computer cannot be afforded by the enterprises. At the same time, the simple computer room environment cannot meet the requirement of system stability. Therefore, an efficient, economical, and easy-to-implement mass data-processing scheme is put forward and applied to seafood product big data processing system that will be of great practical significance.

Distributed storage is an important research topic for the efficient and reliable storage and rapid access to big data resources of seafood products. Meanwhile, in centralized storage, users can directly obtain the history and current status regarding the production of a seafood product and predict complex events and their laws. Based on big data analysis, through centralized data management methods, seafood enterprises can perform seafood production and circulation facilitates mariculture breeding, which can conduct a dynamic assessment of the breeding process, dynamic evaluation for seedling growth and breeding environment, quality assessment of marine food, and the dynamic resource adjustment based on deployment process and order.

The paper is organized as follows. Application scenarios of seafood product big-data are described in Section 2. The storage optimization of seafood product big data is described in Section 3. Section 4 describes the parallel processing method of seafood product big data. The verification and analysis are reported in Section 5. Finally, in Section 6, the conclusions are reported.

2. Application Scenario of Seafood Product Big Data

In a monitoring system for seafood mariculture and traceability system for production and circulation processes, a large amount of data acquisition nodes transfer data to the data center, forming considerable heterogeneous data flow. In this transfer process, a data center platform not only must enable the reliable storage of heterogeneous data but also must allow timely query, retrieval, analysis, and processing of big data. The characteristics of seafood product big data are analyzed [7–12] as follows:

(1) Great capacity of data: the capacity increases from TB level to PB level. Monitoring and traceability systems comprise structured data and unstructured data (e.g., images and videos). Big data technologies are capable of saving large pieces of information in their most primitive state, thereby effectively guaranteeing data integrity.

(2) Correlation among multiple heterogeneous data: inner correlations exist in a wide variety of data. In the processes of big data applications, the correlation analysis of multiple heterogeneous data is essential to the various heterogeneous data and is essential to outside data of a seafood mariculture production system and the cultivation, mariculture, fishing, and production data. Such correlation analyses increase the complexity of advanced applications, such as farming condition assessment, quality evaluation, and sales forecast.

(3) Temporal and spatial attributes of data query: a terminal data acquisition node has a spatial attribute of geographical position, and a data acquisition sample has a time attribute. The multicondition query of seafood big data can be conducted based on complex logical constraints.

(4) Data-processing speed meets real-time requirements: the advanced applications, such as breeding condition assessment, quality assessment, and sales forecasts, require off-line analysis and processing when historical data are massive. Thus, a data-processing platform is necessary, and it provides parallelization batch-processing capability to a large amount of historical data.

(5) The low value of big data: for instance, in video data monitored continuously in mariculture and seafood production scenes, its valuable length of time may be only 1–2 seconds. When process and quality tests are performed by hand or based on experience, only a small amount of abnormal data are analyzed and adopted, and some normal data are abandoned. However, a large number of normal data may be an important basis for problem analysis and judgment, and valuable information may be mined from these data for the translation of information into knowledge and discover rules. The knowledge facilitates correct decisions and actions.

Because of the significant challenges in storing and managing seafood product big data, data management technology based on cloud computing is known to be an effective method. In many cloud-computing technologies, the Hadoop of the open-source Apache community [13–18] is an open-source framework with scalability and high fault tolerance. Hadoop distributed file system (HDFS) and parallel programming framework Hadoop MapReduce [19], which are efficient in big data storage and processing and suitable to distributed storage and management of all types of resource data in ordinary computers, provide high data throughput for applications, which are appropriate for running applications with large data sets. They have been widely used for mass data storage and analysis in Yahoo and Facebook [20, 21]. Therefore, the Hadoop cloud-computing technology is an important option for the storage and processing of seafood big data resources.

3. Storage Optimization of Seafood Product Big Data

3.1. Strategy for Big Data Storage and Distribution

3.1.1. Problems of Data Storage and Platform Optimization. Existing management systems for seafood product data processing commonly use relational databases for data storage. The data storage models of relational databases store data byline and are designed for data recording and online transaction processing (OLTP). In loading and querying of big data, the performance decreases and cannot satisfy the real-time processing demand and quality monitoring of seafood big data. Current data management methods upload only a small amount of data to the main station system [22, 23], and a large number of data with potentially significant values have been discarded which results in a significant waste of data resources. Infrequently updated cases of monitoring data for process and quality, the data-processing efficiency of databases is extremely low because of the OLTP. The application of cloud-computing technology in the field of mariculture remains in the exploratory stage; the research contents of related literature mainly focus on the design of system architecture, model and data-processing platform, the management, and storage of the big data methods based on the distributed computing technology and the big data storage platform solutions were built-in references [24–30]; and the research of cloud-computing platform optimization using internal relations of seafood product big data resources is relatively limited.

3.1.2. Data Distribution Strategy. The data distribution strategy is a key issue that directly affects query performance in the parallel query processing of distributed data storage systems. Many scholars have conducted considerable research on the data distribution strategy and proposed many parallel data distribution methods, in which the hash algorithm is widely used. The consistent hashing algorithm is used to ensure that each node performs the same sequence of operations in the distributed system, and the consistency algorithm can be executed on each instruction to ensure that the final data of each node are consistent. Paxos is an algorithm used to solve the consistency problem. If there is more than one node, there will be a problem with communication between nodes. In addition, the consistent hashing algorithm is proposed and widely used in all distributed data systems for the minimization of node failure or data migration when the node is increasing. The consistent hashing algorithm provides a reference for this research topic. In the point of view of quality evaluation, breeding status assessment, sales forecasting, and other advanced applications based on big data analysis, a strong correlation exists among some monitoring data that are used in a computing task. The data correlation can be used as the basis for data storage and distribution for disruption of the migration among nodes when data are used and improvement of data query performance.

Given the storage, fast processing, and other issues in seafood product big data management, the Hadoop platform has considerable optimization spaces, and it is selected as a storage and processing platform for seafood product big data according to its characteristics and the internal relations among data. The platform architecture is shown in Figure 1. It has three layers. They are application interface layer, basic management layer, and storage layer.

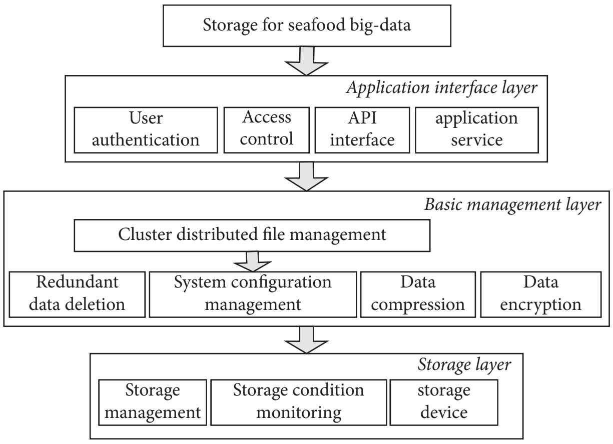

Figure 1: The storage platform architecture of the seafood product big data.

3.2. Storage Optimization of Seafood Product Big Data

3.2.1. Method of Distribution Optimization for Big Data. According to the application demand, the influence factors of seafood data distribution are as follows: (1) data should be distributed to all nodes in a cluster for load balance. (2) The cluster node failure issue, as a normal state in Hadoop, should be considered in data distribution optimization. (3) The copy redundancy measure ensures data reliability and retrieval efficiency. (4) Important factors that affect the overall system performance in a Hadoop operating environment are network transmission and input/output (I/O) operation. Reducing the amount of communication data can effectively improve the efficiency of data processing. As a common data association query in a seafood data management system, the data connection operation is executed in the Reduction stage in the implementation of a Map Reduce-related query, without considering data connection and using the standard distribution strategy of Hadoop. In the Map stage, all data are sorted by grouping in several nodes, and then, the TaskTracker node of the reduction task pulls data through remote access. Many data that do not connect with the data connection operation are copied and transmitted in the process of data connecting. In the process of uploading tracking data, the same source data are stored in the same node according to the source attribute of data. This process can complete the data connection operation in the Map stage, reduce data communication, and improve the overall implementation efficiency of the data association query.

A data correlation-based multireplica consistent hashing algorithm (CMCHA) for the data distribution optimization of Hadoop is proposed by analyzing the affecting factors of data distribution. The basic idea of the storage algorithm is that relevant data are stored centrally, and the main work is executed in the map end for the reduction of data communication load and improvement of the overall performance of data retrieval and analysis. Each type of tracking data may have a different data type and format but has specific collection time and place, which are common keywords for the data retrieval and analysis. Hadoop stores data as three copies. The storage strategy is that one copy is on the local node, one on another node of the same rack, and one on a node of a different rack. Therefore, if local data are damaged, then the node can acquire data from the adjacent node in the same rack. This process is faster than the acquisition of data from a node in the cross rack. If the network of an entire rack is abnormal, data on the other node of the rack are found. The HDFS program attempts to read the nearest copy to reduce the overall bandwidth consumption and reading delay. If a copy exists on the same rack, then the reading program reads it. If an HDFS cluster exists across multiple data centers, then a client firstly reads a copy of the local data center. The storage algorithm considers three aspects: the correlation of data acquisition location, the correlation of data acquisition time, and the correlation of custom data. The first copy data are subjected to hash mapping according to the number of acquisition locations, the second copy data are subject to hash mapping according to the acquisition time stamp, and the third copy data are subjected to hash mapping according to the custom correlation coefficient. The consistent hash algorithm is used in the process, and this process meets the different needs of data query and analysis. The algorithm needs to construct a hash ring. The configuration process is shown in Figure 2.

Step 1: the correlation coefficient of the tracking process data and the number of redundant copies are predefined by the configuration file. The defined number of redundant copies is three.

Step 2: the hash value of each data node in a cluster is calculated and configure in the 0 to 232 hash loop.

Step 3: the hash value of data is calculated based on the temporal and spatial properties of the tracking process data and the correlation coefficient and mapped to the hash loops.

In Figure 2, there are multiple copies of the data in the cloud platform. For the first copy of the data ①, calculating the hash value 1 is according to the source of the data, namely, the monitoring device ID, and mapping it to the hash ring. For the data ②, its hash value 2 is calculated and mapped to the hash loop based on the time attribute of the monitoring data. For the data ③, the hash value 3 is calculated according to the correlation coefficient of the data and mapped to the hash ring. If higher storage reliability is required and more than 3 copies are configured, the hash value i is calculated alternately according to the above three steps and then mapped to the hash ring.

Step 4: the storage location of the data is determined according to the hash value of the data and the data node. In Figure 2, the data are mapped to the nearest data node (e.g., data ① are set to node A).

Step 5: if the node space is insufficient or the node space occupancy exception occurs, the node is skipped to find the next storage node.

3.2.2. Size Optimization of Data Blocks. Two methods are used to block files and store them in HDFS. (1) Each block is stored as a file, and the file name is directly used as the index key of the block data. In addition, (2) all blocks that belong to the same large file are stored as one file, all blocks of documents are stored in different distributed storage nodes, and the stored copy rate is set based on the Hadoop distributed storage strategy, which needs to define and maintain index keys of data blocks. When the data set is large, method (1) produces a large number of small files, which increases the burden on the file system, whereas method (2) does not have such a problem.

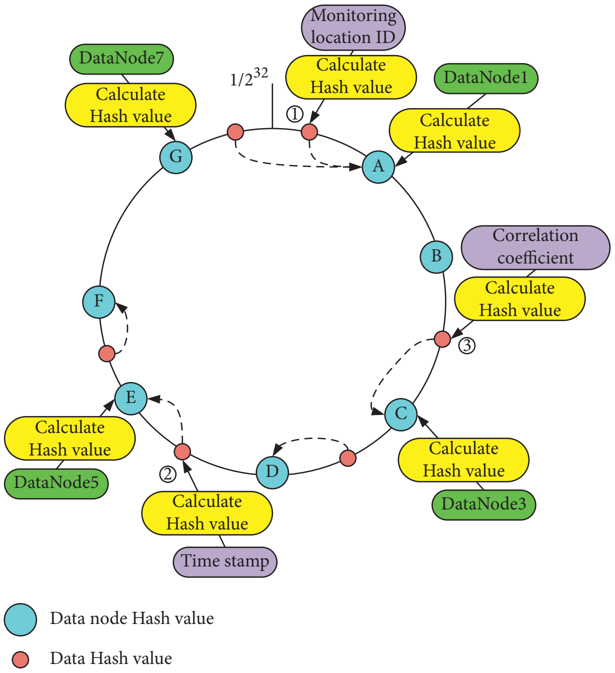

FIGURE 2: The CMCHA flow.

In method (2), for every file, each block of data is stored in HDFS by <key, value> record type, recorded as < Block-ID, MetaData>. Block-ID is the serial number identification of the data block, and MetaData is the binary data of the block. The data type of the record is <int, byte[]>. The byte data of a file block can be determined by the Block-ID. The block storage method of a large file is shown in Figure 3.

The design goal of HDFS is to store large files. The default data block size is 64 MB, which is greater than the size (512 B) of the physical disk block size. The file access time of HDFS mainly includes the system addressing time and data transmission time. The file transfer efficiency fx4 is as follows:

$$\begin{aligned}\eta_{\text{effect}} &= \frac{t_t}{t_t + t_s} \\ &= 1 - \frac{t_s}{S_{\text{block}}/v + t_s},\end{aligned} \tag{1}$$

where t_t is the data transmission time and $t_t = S_{\text{block}}/v$, t_s is the system addressing time, S_{block} represents the data block size, and v represents the data transmission speed.

Equation (1) indicates that n_{effect} is less than 1. The data distribution and index method are usually determined, t_s and v are determined, and S_{Block} should be increased with the increase in n_{effect}. In HDFS, the size of data block S_{block} is configured by the parameter dfs.block.size. A large data block size leads to a reduction in load balance. Therefore, considering the data transmission rate and load balance, the size of the data block is adjusted according to the data scale into the system.

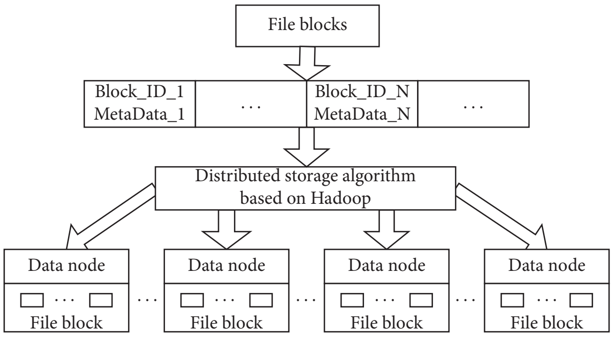

Figure 3: The block storage process of a large file.

4. Parallel Processing Method of Seafood Product Big Data

4.1. Parallel Join Query Algorithm of Multiple Data Sources. A management system of seafood-tracking data conducts comprehensive retrieval for multiple monitoring points and related parameters according to the monitoring location ID, sampling time, and other conditions. The comprehensive retrieval involves the location data (data acquisition equipment name, operation time, acquisition position, etc.), sea geographic information (name, area, location, altitude, latitude, and longitude of mariculture sea), environmental data (environmental temperature and humidity), and production data (fishing time, batch, number, etc.), which requires a data connection for different data sources. The data sets come from the quality information traceability system of the seafood company. The multisource data come from different files. Three data files are connected in the retrieval of a quality control parameter in the production process of dry sea cucumber. The first file is the processing data file, its format is shown in Table 1, and its sampling batch is the product batch code with the coding rules in Reference [1]. Through the coding rules, the product life cycle information could be identified. The second file is the detection data file of the quality control parameter, its format is shown in Table 2, and its sampling information includes moisture, salt, and protein contents (%). The third file is the testing environmental file shown in Table 3, in which the detection position code represents "workshop–section–team–station." The query generates the comprehensive detection data between 9:00 and 9 : 20 on March 14, 2020. The quality data list with the location and environmental information is obtained, which requires the connection of three data files for the acquisition of a data list that satisfies the requirements of the comprehensive query. The connection results are shown in Table 4.

Table 1: The processing data file.

Position ID	Collection time	Sample batch
DL082	2020-03-14 9:08	202003140103100
DL083	2020-03-14 9:18	202003140103300
DL081	2020-03-14 9:10	202003140103200

Table 2: The quality inspection data file.

Position ID	Collection time	Sample batch	Sample information
DL083	2020-03-14 9:10	202003140103200	31.9/7.58/50.2
DL081	2020-03-14 9:00	202003140103300	32.5/7.55/50.5
DL082	2020-03-14 9:08	202003140103100	32.2/7.57/50.4
DL082	2020-03-17 9:00	202003140103100	11.7/10.1/62.2

Table 3: The detection position data file.

Position ID	Address	Collection time	Temperature	Moisture
DL081	ZZ01-I-02-B	2020-03-14 8:00	39	65
DL082	ZZ01-I-02-A1	2020-03-14 8:10	38	63
DL083	ZZ02-II-01-A1	2020-03-14 8:11	38	59

The parallel filtering mapping connection retrieval algorithm is designed according to the demand of data retrieval and format description. The retrieval algorithm is executed in the map end. The main idea of the algorithm is that the data filtering and connection complete in the Map stage and avoid the retrieval operation in the Reduction stage for the saving of network traffic transmission and the improvement of retrieval efficiency. The multireplica consistent hashing algorithm based on data correlation is used for the data distribution, which gathers data from the same data node. The algorithm flow is as follows:

(1) The data are filtered according to the query conditions, and the data that do not satisfy the conditions are eliminated.

(2) The data connection group key, including the detection position ID, timestamp, or correlation coefficient, is determined according to the demand of connecting retrieval.

(3) The data file name is a label to marks the record of each data source.

(4) The records with the same attribute value are divided into a group according to the connection group key, and the data connection is conducted.

After the data optimization distribution, in the mapping process of connection retrieval, filtering, label set, the grouped scheduling, and connection operation are conducted in the local node. The flow of the data optimization distribution and the data connection mode is shown in Figure 4. After data distribution is optimized, operations such as filtering, marker setting, grouping sorting, and connection are performed on the local node during connection query mapping, and connection query results are output to the HDFS file system.

Table 4: The results of date join.

Position ID	Installation address	Collection time	Temperature	Moisture	Sample batch	Sample information
DL081	ZZ01-A-026-B	2020-03-14 9:00	39	65	202003140103300	32.5/7.55/50.5
DL082	ZZ01-A-026-A1	2020-03-14 9:08	38	63	202003140103100	32.2/7.57/50.4
DL083	ZZ02-B-017-A1	2020-03-14 9:10	38	59	202003140103200	31.9/7.58/50.2

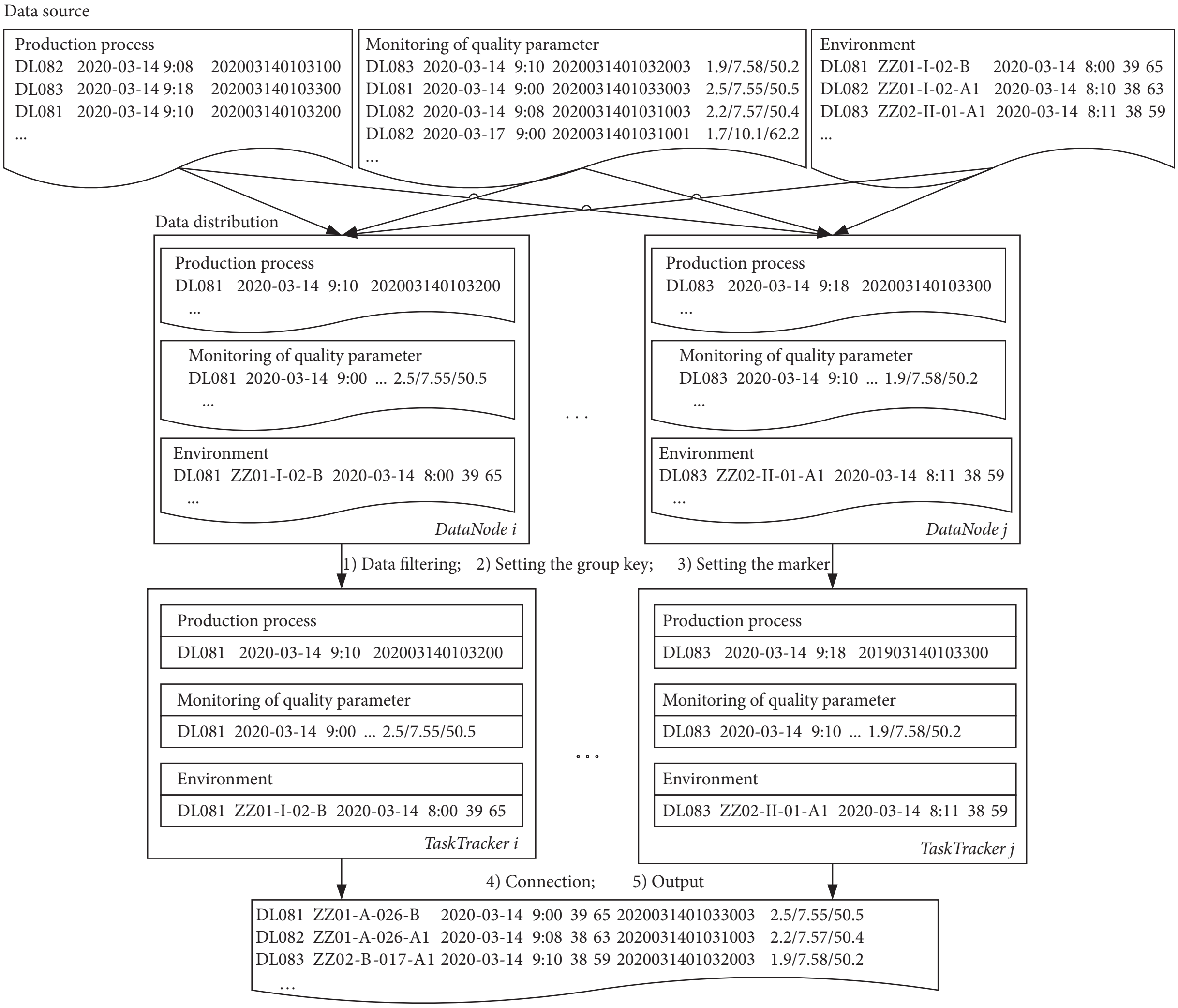
Data source
Production process
DL082 2020-03-14 9:08 202003140103100
DL083 2020-03-14 9:18 202003140103300
DL081 2020-03-14 9:10 202003140103200
...
Monitoring of quality parameter
DL083 2020-03-14 9:10 2020031401032003 1.9/7.58/50.2
DL081 2020-03-14 9:00 2020031401033003 2.5/7.55/50.5
DL082 2020-03-14 9:08 2020031401031003 2.2/7.57/50.4
DL082 2020-03-17 9:00 2020031401031001 1.7/10.1/62.2
...
Environment
DL081 ZZ01-I-02-B 2020-03-14 8:00 39 65
DL082 ZZ01-I-02-A1 2020-03-14 8:10 38 63
DL083 ZZ02-II-01-A1 2020-03-14 8:11 38 59
...
Data distribution
Production process
DL081 2020-03-14 9:10 202003140103200
...
Monitoring of quality parameter
DL081 2020-03-14 9:00 ... 2.5/7.55/50.5
...
Environment
DL081 ZZ01-I-02-B 2020-03-14 8:00 39 65
...
DataNode i
...
Production process
DL083 2020-03-14 9:18 202003140103300
...
Monitoring of quality parameter
DL083 2020-03-14 9:10 ... 1.9/7.58/50.2
...
Environment
DL083 ZZ02-II-01-A1 2020-03-14 8:11 38 59
...
DataNode j
1) Data filtering; 2) Setting the group key; 3) Setting the marker
Production process
DL081 2020-03-14 9:10 202003140103200
Monitoring of quality parameter
DL081 2020-03-14 9:00 ... 2.5/7.55/50.5
Environment
DL081 ZZ01-I-02-B 2020-03-14 8:00 39 65
TaskTracker i
...
Production process
DL083 2020-03-14 9:18 201903140103300
Monitoring of quality parameter
DL083 2020-03-14 9:10 ... 1.9/7.58/50.2
Environment
DL083 ZZ02-II-01-A1 2020-03-14 8:11 38 59
TaskTracker j
4) Connection; 5) Output
DL081 ZZ01-A-026-B 2020-03-14 9:00 39 65 2020031401033003 2.5/7.55/50.5
DL082 ZZ01-A-026-A1 2020-03-14 9:08 38 63 2020031401031003 2.2/7.57/50.4
DL083 ZZ02-B-017-A1 2020-03-14 9:10 38 59 2020031401032003 1.9/7.58/50.2
...

Figure 4: The data optimization distribution and map data connection mode.

4.2. Parallel Extraction Algorithm of Multichannel Data Fusion Features. A seafood big data traceability system collects and preserves multichannel data sequences. The dynamic relationship among the data sequences implies considerable feature information that reflects the status of mariculture, seafood production, and circulation. Academics have proposed various evaluation methods for the dynamic relationship in different application backgrounds [31]. In [32, 33], the multiscale multivariate entropy (MSMVE) analysis method was proposed. This method evaluates the dynamic relationship of multichannel time series according to complexity, interprediction and longtime relevance, and inherent nonlinear coupling characteristics. The method has been applied in many fields of physics, physiology, and other disciplines and has shown its potential value in theory and application [34].

In this study, MSMVE is applied to the data fusion feature extraction for quality detection data in a six-channel synchronous acquisition. When the mass data are reached, the MSMVE algorithm runs slowly. Therefore, a parallel MSMVE algorithm is designed in the Hadoop platform based on CMCHA for the improvement of data fusion feature extraction speed.

The information synchronously acquired from the six-channel equipment is stored in six files. For parallel data analysis, the data sections are uploaded and stored to the HDFS of the standard Hadoop platform. Each data section has a timestamp and randomly distributes in multiple data nodes according to the rack-sensing strategy. The multichannel information used in an MSMVE calculation task is assigned to different nodes because of disregarding the data correlation. Thus, the computing model of the parallel MSMVE algorithm is filtering the data in the mapping process. The signal is sent to the reduction end through the network. The solution of the MSMVE is obtaining in the reduction process. The data distribution and parallel computing process are shown in Figure 5.

Each channel file is divided into n segments distributed and stored on several data nodes. The information segments of synchronous acquisition and the same timestamps are used in the same MSMVE computing tasks and marked in the figure with small circles with the same number. The MSMVE calculation is completed in the Reduction stage. The results are output to the HDFS and saved there, as marked by small squares with numbers in Figure 5.

The multichannel data distribution is optimized by CMCHA. The temporal correlation of data and the collection time stamp are considered keywords to calculate the hash storage location. The synchronization data are gathered by optimizing the data distribution, and the MSMVE computing tasks are completed at the Map stage. The data distribution and parallel computing process are shown in Figure 6.

The flow of the MSMVE parallel feature extraction algorithm based on CMCHA is as follows:

(1) According to the time of the MSMVE calculation task, the data are filtered, the data distribution is optimized, and the data that do not meet the time condition are eliminated.

(2) The collection timestamp is set as the main connection group key, and each record is marked.

(3) Records with the same attribute value are assigned to a group by the connecting group key. The MSMVE algorithm is implemented.

(4) The calculation results of the MSMVE are output to HDFS.

The flow of the MSMVE algorithm is as follows:

(1) The original p-dimensional (channel) time sequences are $\{x_{k,i}\}_{i=1}^{N}$, $k = 1, 2, \ldots, p$. Each dimension sequence has N points. The multivariable and coarse-grained time sequences $\{y_{k,j}^{\varepsilon}\}$ are constructed for preset scale factor ε, that is,

$$y_{k,j}^{\varepsilon} = \frac{1}{\varepsilon} \sum_{i=(j-1)\varepsilon+1}^{j\varepsilon} x_{k,i}, \quad k = 1, 2, \ldots, p, \ 1 \le j \le \frac{N}{\varepsilon}. \tag{2}$$

If $\varepsilon = 1$, then $\{y_{k,j}^{\varepsilon}\}$ is the original time sequence.

(2) The parameter vector $M = [m_1, m_1, \ldots, m_p]$ and the time-delayed vector $\tau = [\tau_1, \tau_1, \ldots, \tau_p]$ are predetermined and embedded based on the multivariable time sequence model. The $(N - n)$ complex delayed vectors $Y_m(i)\,(m = \sum_{k=1}^{p} m_k)$ are established as

$$Y_m(i) = \left[y_{1,i}, y_{1,i+\tau_1}, \ldots, y_{1,i+(m_1-1)\tau_1}, y_{2,i}, y_{2,i+\tau_2}, \ldots, y_{2,i+(m_2-1)\tau_2}, \ldots, y_{p,i}, y_{p,i+\tau_p}, \ldots, y_{p,i+(m_p-1)\tau_p}\right], \quad i = 1, 2, \ldots, N - n, \ n = \max\{M\} \times \max\{\tau\}. \tag{3}$$

(3) The distance between $Y_m(i)$ and $Y_m(j)$ is defined as

$$d[Y_m(i), Y_m(j)] = \max_{l=1,2,\ldots,mn}\{|x(i + l - 1) - x(j + l - 1)|\}. \tag{4}$$

(4) For the given threshold r, the probability of event $P_i[Y_m(i), Y_m(j) \le r\,(j \ne i)]$ is calculated as $B_i^m(r) = P_i/(N - n - 1)$, which means the correlation degree between $Y_m(j)\,(i \ne j)$ and $Y_m(i)$ and indicates the regularity of the sequence $\{Y_m(j)\}$.

(5) The average of probability $B^m(r)$ for all i is calculated as

$$B^m(r) = \frac{1}{N - n} \sum_{i=1}^{N-n} B_i^m(r). \tag{5}$$

(6) m is extended to $m + 1$ in Step 2, Steps 3 to 5 are repeated, and $B^{m+1}(r)$ is generated.

(7) The multispecification and variable sample entropy are calculated as

$$\text{MSMVE}(M, \tau, r, N) = -\ln \frac{B^{m+1}(r)}{B^m(r)}. \tag{6}$$

4.3. Performance Analysis of the Algorithm. Each data file is divided into blocks according to the fixed size specifications on the standard Hadoop platform that is not optimized, and the storage distribution is conducted by randomly selecting nodes and the data block as a unit. Therefore, in the process of data analysis, the data are randomly distributed on different data nodes, and the nodes need much data communication when using the MapReduce framework for parallel computing. The CMCHA uses the inner relationship among state-monitoring data for data distribution optimization and data aggregation, which reduces data communication among data nodes in the process of data analysis. When data are not fully stored, the network data communication in running the parallel analysis program is analyzed below.

The number of Hadoop cluster nodes is assumed to be N, and the parallel analysis task is divided into M subtasks

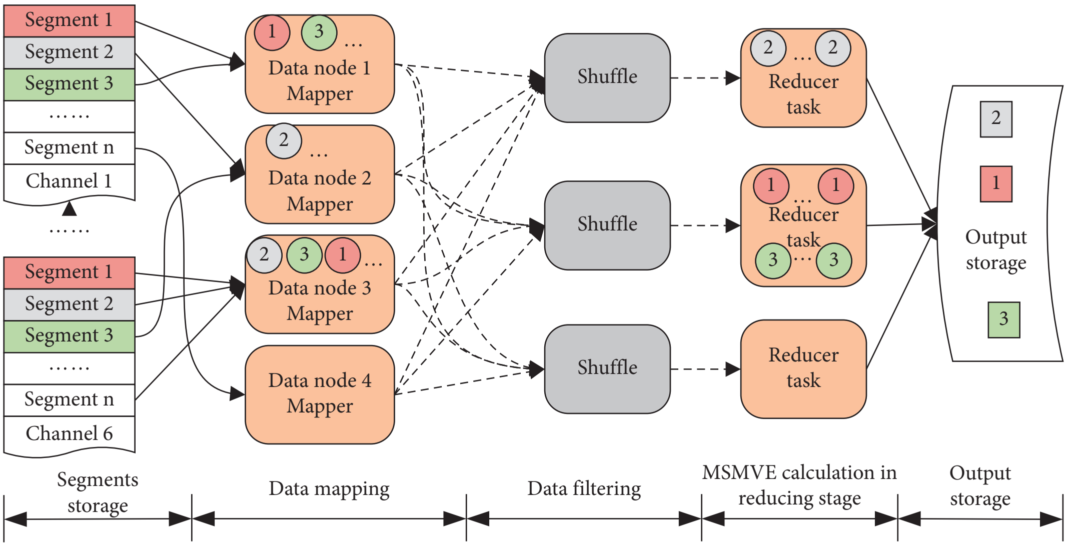

Figure 5: The data distribution on the standard Hadoop platform and reduction feature extraction.

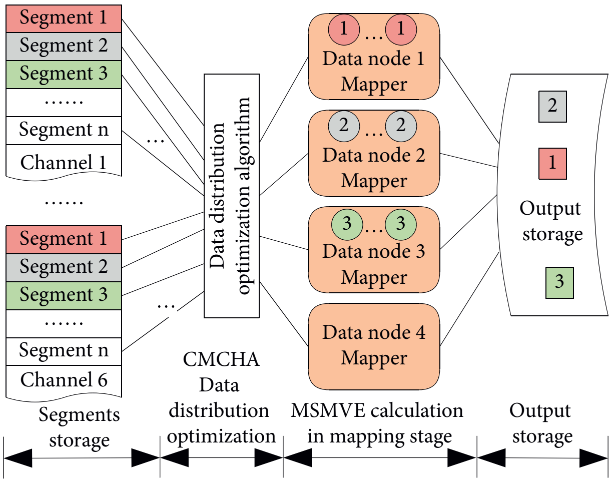

Figure 6: The data distribution optimization with CMCHA and Map feature extraction.

Map_{ij} (i is the task ID, $1 \le i \le M$, and j is the node ID). The single data block size is fixed to d bytes. The number of blocks used by the subtask Map_{ij} executed on node j is a_i. The copy number of the data is R. The task scheduler of MapReduce ensures the local property of data; thus, data blocks needed by Map_{ij} are on j node, and the worst case is only one data block in local. Hadoop does not put multiple copies of the same data blocks on the same node. Therefore, the ratio that only one data block needed by Map_{ij} is in local is represented as

$$p_{i,1} = \frac{a_i P_{N-1}^{R-1} R \left(P_{N-1}^{R}\right)^{a_i-1}}{\left(P_N^R\right)^{a_i}} = \frac{a_i R}{N^{a_i}\left(N-R+1\right)^{a_i-1}}. \tag{7}$$

The number of data blocks needed to be pulled is $a_i - 1$. The ratio that k data blocks needed by Map_{ij} are in local is as follows:

$$p_{i,k} = \frac{C_{a_i}^k \left(P_{N-1}^{R-1}\right)^k R^k \left(P_{N-1}^{R}\right)^{a_i-k}}{\left(P_N^R\right)^{a_i}} = \frac{C_{a_i}^k R^k}{N^{a_i}\left(N-R+1\right)^{a_i-k}}. \tag{8}$$

The number of data blocks needed to be pulled is $a_i - 1$. The ratio that a_i data blocks needed by Map_{ij} are in local is $p_{i,\text{all}} = (R/N)^{a_i}$. Data need not be pulled on other nodes, and no data communication occurs. Hence, when Map_{ij} is executed, the number D_i of data blocks may be expressed as probability average.

$$\begin{aligned} D_i &= p_{i,1}(a-1) + p_{i,2}(a-2) + \cdots + p_{i,k}(a-k) + \cdots + p_{i,a_i-1} \\ &= \sum_{k=1}^{a_i}\left[p_{i,k}\left(a_i-k\right)\right] = \sum_{k=1}^{a_i}\left[\frac{C_{a_i}^k R^k}{N^{a_i}\left(N-R+1\right)^{a_i-k}}\left(a_i-k\right)\right]. \end{aligned} \tag{9}$$

The total amount of data communication is expressed as

$$D = \sum_{i=1}^{M}\sum_{k=1}^{a_i}\left[\frac{C_{a_i}^k R^k}{N^{a_i}\left(N-R+1\right)^{a_i-k}}\left(a_i-k\right)d\right]. \tag{10}$$

The network bandwidths that subtask Map_{ij} pulls data from different data nodes are different in the process of data communication. In the Hadoop cluster network topology, the network communication bandwidth among nodes in the same rack is c_1, the network communication bandwidth among nodes in different racks in one room is c_2, and the network communication bandwidth among nodes in different rooms is c_3. When the data block required by subtask Map_{ij} is not in local, the rate that the data block and Map_{ij}

are located in the same frame is P_1, located in the same room and different racks are P_2, and located in different rooms is P_3. The average network communication bandwidth is $C_{avr} = \sum_{j=1}^{3} P_i c_i$. The execution time of the parallel algorithm task is as follows:

$$T = \frac{D}{C_{avr}} = \frac{\sum_{i=1}^{M} \sum_{k=1}^{a_i} \left[\left(C_{a_i}^{k} R^{k} / N^{a_i} \left(N - \ R + 1 \right)^{a_i - k} \right) \left(a_i - k \right) d \right]}{\sum_{j=1}^{3} p_i c_i}. \qquad (11)$$

Equation (11) implies that the main factors, which affect the performance of the parallel algorithm, include the size of the Hadoop cluster, the number of data storage copies, the size of the data block, the probability of data locality, and the communication network bandwidth among nodes. In the case of random distribution of data, the greater the number N of nodes in the Hadoop cluster is, the worse the aggregation of the relevant data is; the amount of network data communication increases; and the poorer the algorithm performance is.

The data replica strategy can improve the probability of data locality for the enhancement of algorithm performance. However, because of the system data capacity, data consistency, and other restrictions, the copy number R of data cannot be too large, generally, not more than three copies. Equation (10) presents that the quantity of data communication is proportional to the size of the data block. When the data block is small, the communication quantity is small. Nevertheless, the preceding analysis implies that the size of the data block is proportional to the data transmission rate, so the size of the data block must be comprehensively considered according to the file transmission rate, load balance, and other various factors. When the scale N of the Hadoop cluster increases constantly, the number of copies and the data block size cannot be adjusted freely, and the usage of a data distribution optimization algorithm for related data gathering and local data-processing improvement can effectively enhance the parallel algorithm performance.

5. Verification and Analysis

Aiming at the two core problems of big data processing, reliable storage, and fast access of big data, the Hadoop cloud-computing experimental platform was established, and the above algorithms were tested and run on the cluster built in the laboratory.

5.1. Establishment of Hadoop Platform. The Hadoop cluster platform, which consists of 10 nodes that are 10 servers, is set up in the laboratory. A node is configured with an Intel Core i5 CPU having 4 cores, 2.6 GHz frequency, 8 GB memory, and 1 TB hard disk. The interconnection of cluster nodes uses Gigabit Ethernet. One node is specified as NameNode, and a different node is specified as JobTracker in the cluster. These nodes are the main control nodes. The remaining nodes are the clients, as DataNode and TaskTracker. The operating system is Ubuntu, and the deployment is VMware. In VMware, an Ubuntu virtual machine is installed, and the other two virtual machines are exported or cloned. The connection of the virtual machines is bridge connections, which ensures that the IP of the virtual machine and the IP of the host are in the same IP segment, and thus, the virtual machine and the host can communicate with each other. The specification of the data block is set as 64 MB, each node is assigned three mapping calculation task slots and one approximately reduced computation task slot, and the total number corresponds to four CPU cores. The overall I/O performance of a cluster is tested by using TestDFSIO [35].

5.2. Performance Verification of the Parallel Join Query Algorithm for Multiple Data Sources. For testing the performance of the join query algorithm for multiple data sources after storage and distribution optimization, a comparative analysis is conducted for the algorithm and join query processing algorithm in the reduction end provided by the current stable version of the Hadoop platform (Version 1.0.4). With the progress of data acquisition and processing technology, data uncertainty is widespread. Uncertainty data can be expressed in various forms, such as relational data, semistructured data, stream data, or mobile object data. Uncertainty is introduced when data are automatically extracted from unstructured or semistructured data sources. Uncertainty is also introduced when data are retrieved from unreliable or outdated sites. A real data set stored in the seafood big data traceability system of the laboratory is used in the test. The data set is shown in Table 5. This test mainly verifies the performance of the proposed algorithms without considering the uncertainty of the data. The uncertainty with more detail in the approach will be carried out in the subsequent studies, including the data model, data preprocessing and integration, storage, and indexing, query processing, and so on.

The factorial design of experiments is a mathematical-statistical method to arrange experiments and analyze experimental data. The whole-factor design of experiments refers to a design, in which all combinations of all levels of factors are performed at least once. The running time is used to validate the retrieval query performance of the algorithm. The factors, which affect the running time, include the size of the subset, namely, the number of data records, the time, and location of data acquisition. According to references [36–38], the number of the data records is determined as the main affecting factor through the MINITAB experiments design tools. The number is set that the low level is 100,000 and the high level is 13.76 M. The following algorithm performance verification is performed.

(1) The Execution Time Variation Trend of the Parallel Join Query Algorithm for Multiple Data Sources. To verify the algorithm performance for multiple data sources based on CMCHA under different query conditions, we selected three typical join query requirements to run the test and record the running time of the parallel join query algorithm. The three join query requirements are as follows: (1) full join query, in which no query condition is set and the position ID connects

Table 5: The data set for the join query.

Filename	Copy number	File size	Total size	Record number
Production process	3	627 kB	1881 kB	1910
Monitoring of quality parameter	3	370 GB	1110 GB	13.62 M
Monitoring environment	3	215 MB	645 MB	4175

detection position, production process, and quality inspection data for the query of equipment information; (2) device as query condition, in which the information of the device ID is queried and monitored within a certain range; and (3) time as query condition, in which information in the defined time range is queried and monitored. The SQL descriptions of these three typical queries are shown in Table 6.

In the course of verification, a subset of different scales is selected from the data set, from 100,000 records to the data set (13.76 million records). The query result of the same query condition is invariable so to avoid the contingency in the experiment and reduce the experimental error, run 10 times under each query condition, and take the average running time of 10 times. The relationship between the execution time of the multidata source parallel connection query algorithm based on CMCHA and the data size is shown in Figure 7. With the growth of data size, the execution time of data retrieval increases slowly. The data storage layout is optimized by CMCHA, and the integrated query is completed in the mapping process. The network communication is effectively reduced, and the stability of the query performance is ensured.

(2) The Execution Time Comparison of Data Join Query. Full join query and join query based on device or time query condition are conducted by using the parallel connection query algorithm of multiple data sources according to CMCHA and connection-query-processing algorithm in the reduction end on the standard Hadoop platform for data set(13.76 M). To avoid contingency in the experiment and reduce the experimental error, each algorithm runs 10 times under the same conditions and takes the average value. The execution time comparison results are shown in Figure 8. The execution times of the former algorithm under the three query conditions are 33.1%, 32.6%, and 31.9% of the execution time of the latter algorithm. For this improved operation performance, one of the main reasons is that the data from multiple data sources are gathered after data layout optimization, and the other is that the data connection is completed locally in the mapping task, which eliminates the data transmission from the mapping end to the reduction end and reduces the effect of reduction task start on the performance.

5.3. Performance Test of the MSMVE Parallel Feature Extraction Algorithm.

The data set for the test is stored independently in six files, in which one file is approximately 6.5 MB (81920 sampling points). When the number of samples is limited, the data set is replicated such that the performance of the algorithm is ensured when used to process big data. Consequently, the size of one file reaches 650 MB, and the data size reaches 3900 MB.

(1) The Variation Trend of the Execution Time of Data Uploading. For the verification of the effect of optimal storage strategy on data upload speed, the data cluster from the local file system of a client is uploaded to the HDFS of Hadoop by using CMCHA and the default data distribution strategy separately. In the upload process, the data set size increases from one file (650 MB) to six files (3900 MB), the variation trend of execution time is shown in Figure 9. The experimental results indicate that the data transmission time increases linearly with the growth of data size, and the data transmission rate is stable. The optimal storage strategy exerts a minimal influence on the speed of data upload, the execution time of the upload process is slightly longer than that of the random distribution, and the data transmission rate is slightly decreased. The average data transmission rate of CMCHA is 19.7 M/s, whereas the average data transmission rate of the standard Hadoop platform is 21.3 M/s. The main reason for the slight decrease in the data transmission rate is that the data layout optimization needs additional processing time to complete the selection of data nodes, and the standard Hadoop platform adopts the random distribution strategy.

(2) The Variation Trend of the Parallel Execution Time of MSMVE. 5,210 sample points (for 0.5 s data acquisition) are selected as the length of the sample signal to calculate the MSMVE. The multiscale factor ε takes 8 and 15. The embedded dimension vector is M [2], the time delay vector is [1], and the threshold parameter is $r = 0.45$. The data set contains 1,600 samples (approximately 3,900 MB). The scale of the experimental data increases from 200 to 1,600, and the execution time of the algorithm is shown in Figure 10. With the increase in data scale, the execution time of MSMVE increases slowly, and the data-processing speed is increased. Therefore, the algorithm is suitable for processing large-scale data. The MSMVE calculation process is completed in the mapping process, the overall execution time is unaffected by the network communication bandwidth, and the algorithm performance is stable.

(3) The Comparison of the Execution Time of the Feature Extraction Algorithm. Different size sample data sets are selected, and the scale of the experimental data is increased from 200 to 1,600. The execution time of the multichannel data fusion parallel feature extraction algorithm based on CMCHA at the mapping end and the feature extraction algorithm on the standard Hadoop platform in the reduction end are compared. The execution time of the former is only approximately 35% of the execution time of the latter, as

Table 6: The SQL descriptions of the join query.

Query type	Query criteria	Query statement
Full join query	No condition setting	Select location ID, install location, acquisition time, temperature, humidity, sampling batch, sampling information From production process, monitoring of quality parameter, monitoring environment Where production process. Location ID = monitoring of quality parameter. Location ID = monitoring environment. Location ID
Location condition join	Monitoring location as query condition	Select location ID, install location, acquisition time, temperature, humidity, sampling batch, sampling information From production process, monitoring of quality parameter, monitoring environment Where production process. Location ID = monitoring of quality parameter. Location ID = monitoring environment. Location ID and location ID between[ID1,IDn]
Time condition join	Time as query condition	Select location ID, install location, acquisition time, temperature, humidity, sampling batch, sampling information From production process, monitoring of quality parameter, monitoring environment Where production process. Location ID = monitoring of quality parameter. Location ID = monitoring environment. Location ID and collection time between[T1,Tn]

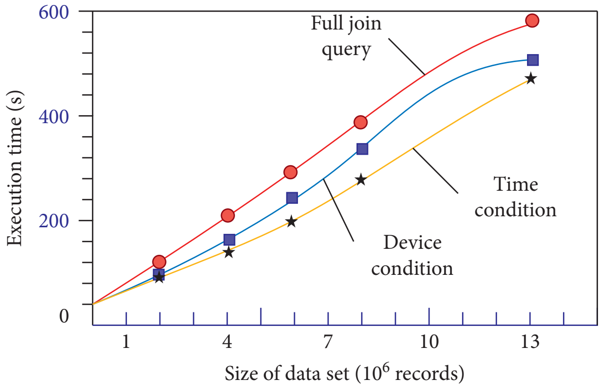

Figure 7: The execution time of data join of multidata sources.

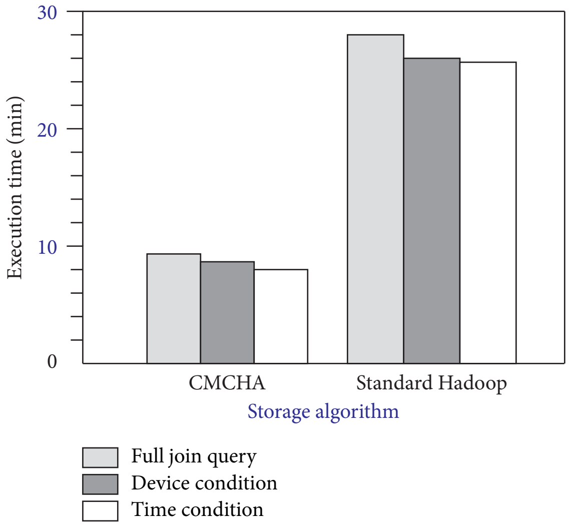

Figure 8: The execution time comparisons of data join of multidata sources.

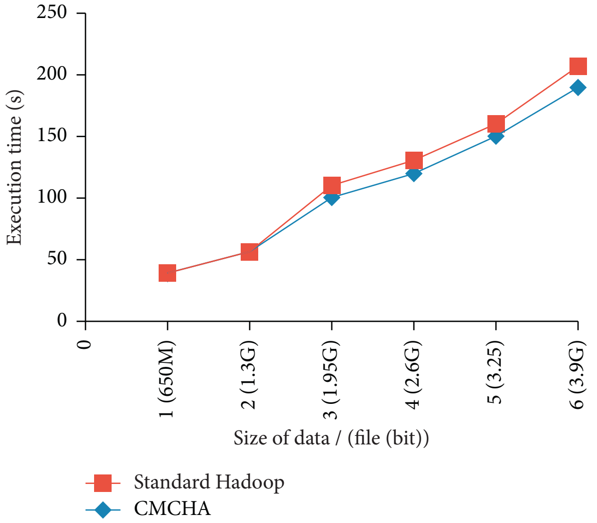

Figure 9: The execution time of data uploading.

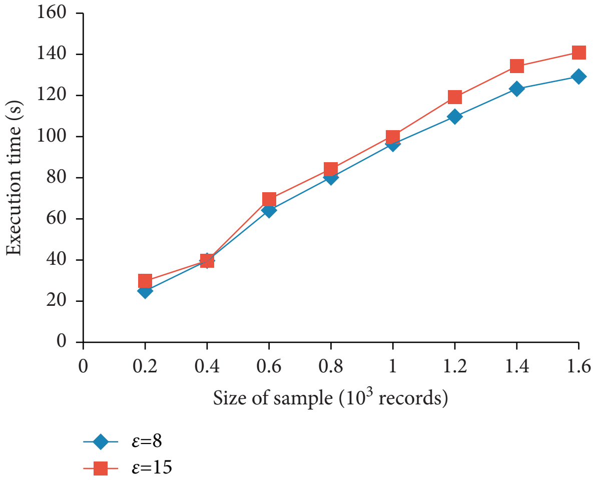

Figure 10: The execution time of MSMVE.

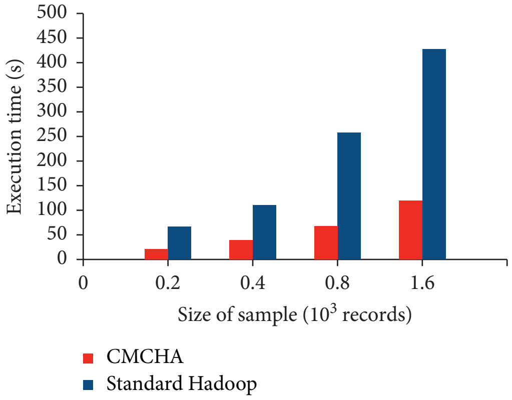

Figure 11: The execution time comparisons.

shown in Figure 11. The main reason is that the data distribution optimization of CMCHA saves considerable time of network communication and the reduction task process.

6. Conclusions

This research studies the unified organization and management of seafood product big data resources based on the Hadoop platform by using a distributed storage optimization and parallel processing method. A multireplica consistent hash data storage algorithm considering data correlation is proposed. The algorithm assembles data correlating a cluster according to product main attribute, timestamp, and correlation coefficient to improve the efficiency of data processing. MapReduce parallel programming is used based on the multireplica consistent hashing algorithm for the realization of a multidata source map join query algorithm and multichannel data fusion feature extraction algorithm for the big data resources of seafood products. The experimental results show that the storage optimization example effectively enhances the execution speed. The execution time of multidata source parallel retrieval is only 32% of the time of the standard Hadoop scheme, and the execution time of the multichannel data fusion feature extraction algorithm is only 35% of the time of the standard Hadoop scheme. In addition, it has room for further improvement in the transmission efficiency of a large file through the experimental study of the data resources of different enterprises and seafood. This research established a foundation to provide the organization, management, sharing, and transmission of massive resources and information services by this platform for enterprises and relevant government departments.

The current data size is only in the GB level because of the limitation of the actual hardware and the data size, but it can reflect the changing trend of the operation time of algorithms with changing data sizes. TB-level data will be validated in the subsequent research for the technical preparation of higher level applications based on seafood product big data. Through the deployment and application in different regions, it is also found that the platform still has room for further improvement in the efficiency of large file transmission.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare no conflicts of interest.

Authors' Contributions

Y.W. and J.L. formally analyzed the data; Y.W. investigated the data; S.C. and X.Z. performed methodology; and J.L. performed software.

Acknowledgments

Thanks are due to all the authors of the references who inspire us and help us. This research was funded by the National Natural Science Foundation of China (71603035) and Natural Science Foundation of Liaoning Province (LJKZ0532 and J2020108).

References

[1] Y. J. Wang, H. Zhang, J. L. Shi, S. Q. Wang, M. J. Zhou, and D. Q. Wang, "Quality information traceability system based on seafood's production process," *Transactions of the Chinese Society of Agricultural Engineering*, vol. 31, pp. 264–271, 2015.

[2] C. H. Sun, X. T. Yang, W. Y. Li, X. X. Liu, and D. L. Li, "Design and realization of distributed traceability system of aquatic products based on supervision mode," *Transactions of the Chinese Society of Agricultural Engineering*, vol. 28, pp. 146–153, 2012.

[3] A. Gorelik, *The Enterprise Big Data Lake: Delivering the Promise of Big Data and Data Science*, O'Reilly Media, Sebastopol, CA, USA, 2019.

[4] B. Y. Cao, H. S. Feng, J. H. Liang, and X. Li, "Hilbert curve and cassandra based indexing and storing approach for large-scale spatiotemporal data," *Geomatics and Information Science of Wuhan University*, vol. 46, no. 5, pp. 620–629, 2021.

[5] D. W. Li, W. J. Huang, J. H. Hu, and Y. Z. Qian, "A distributed redundant real-time data storage mechanism," *Journal of Shanghai Jiaotong University*, vol. 48, pp. 948–952, 2014.

[6] M. M. Hu, Y. Tang, J. Li, and H. S. Chen, "A method for distributed spatial metadata retrieval based on directory services," *Compute.Eng. Sci.*, vol. 33, pp. 162–166, 2011.

[7] Z. H. Liu and Q. L. Zhang, "Research overview of big data technology," *Journal of Zhejiang University*, vol. 48, pp. 957–972, 2014.

[8] J. H. Li, Z. H. Shen, and X. F. Meng, "Scientific big data management: concepts, technologies and system," *Journal of Computer Research and Development*, vol. 54, pp. 235–247, 2017.

[9] S. Aisha, K. Ahmad, and G. Abdullah, "Big data storage technologies: a survey," *Frontiers of Information Technology & Electronic Engineering*, vol. 18, pp. 1040–1070, 2017.

[10] X. Q. Cheng, X. L. Jin, Y. Z. Wang, X. F. Guo, T. Y. Zhang, and G. J. Li, "Survey on big data system and analytic technology," *Journal of Software*, vol. 25, pp. 1889–1908, 2014.

[11] N. O. Krasnoshlyk, "A modified bat algorithm for solving global optimization problem," *Radio Electronics, Computer Science, Control*, vol. 4, pp. 96–103, 2015.

[12] L. H. Zheng, H. Guo, M. Z. Li, X. C. Li, Y. Chen, and C. Y. Xiao, "Grain yield data collection and service for heterogeneous platforms," *Transactions of the Chinese Society of Agricultural Engineering*, vol. 32, pp. 142–149, 2016.

[13] H. Zahid, T. Mahmood, A. Morshed, and T. Sellis, "Big data analytics in telecommunications: literature review and architecture Recommendations," *IEEE/CAA J. Autom. Sinica*, vol. 7, no. 1, pp. 18–38, 2020.

[14] D. Chrimes, M. H. Kuo, A. W. Kushniruk, and B. Moa, "Interactive big data analytics platform for healthcare and clinical services," *Global Journal of Environmental Sciences*, vol. 1, no. 1, 2018.

[15] F. Yang, H. R. Wu, H. J. Zhu, H. H. Zhang, and X. Sun, "Massive agricultural data resource management platform based on Hadoop," *Computer Engineering*, vol. 37, pp. 242–244, 2011.

[16] H. W. Mei, Z. Q. Mi, and G. L. Wu, "Massive data processing of intermittent energy based on MapReduce model," *Automation of Electric Power Systems*, vol. 38, pp. 76–99, 2014.

[17] J. R. Chen and J. J. Le, "Reviewing the big data solution based on Hadoop ecosystem," *Computer Engineering Science*, vol. 35, no. 10, pp. 25–35, 2013.

[18] Y. L. Zhai, Z. Luo, K. Yang, and S. C. Xu, "High performance massive data computing framework based on Hadoop cluster," *Computer Science*, vol. 40, pp. 100–103, 2013.

[19] J. Dean and S. Ghemawat, "MapReduce," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[20] B. F. Cooper, E. Baldeschwieler, R. Fonseca et al., "Building a cloud for yahoo," *IEEE Data Eng. Bull*, vol. 32, pp. 36–43, 2009.

[21] D. Borthakur, J. Gray, J. S. Sarma et al., "Aiyer: Apache hadoop goes realtime at facebook," in *Proceedings of the 38th ACM SIGMOD Int Conf on Management of Data(-SIGMOD'11)*, pp. 1071–1080, ACM, Amsterdam Netherlands, June 2011.

[22] Q. S. Zhao, L. Chen, B. Sun, Y. Zhu, and H. Y. Jiang, "Algorithm implementation and tested of crop growth model based on Hadoop of cloud computing," *Transactions of the Chinese Society of Agricultural Engineering*, vol. 28, pp. 179–186, 2013.

[23] S. C. Wang, L. H. Zheng, M. Z. Li, and K. Zhao, "The development and implementation of farmland gis module based on cloud computing platform," in *Proceedings of the 2011 IEEE International Conference on Computer Science and Automation Engineering (CSAE 2011)*, Shanghai, China, June 2011.

[24] J. P. Qian, X. T. Yang, B. Y. Zhang, X. M. Wu, and B. Xue, "RFID-based solution for improving vegetable producing area traceability precision and its application," *Transactions of the Chinese Society of Agricultural Engineering*, vol. 28, pp. 234–239, 2012.

[25] L. C. Chen, Y. M. Hu, F. Y. Zhang, W. J. Duan, and P. X. Yu, "Performance improving design on cloud computing for agricultural products safety traceability system," *Transactions of the Chinese Society of Agricultural Engineering*, vol. 29, pp. 268–274, 2013.

[26] W. H. Bangyal, J. Ahmad, I. Shafi, and Q. Abbas, "A forward only counter propagation network-based approach for contraceptive method choice classification task," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 24, no. 2, pp. 211–218, 2012.

[27] Y. F. Bo and H. Y. Wang, "The application of cloud computing and the internet of things in agriculture and forestry//IJCSS," in *Proceedings of the 2011 International Joint Conference on Service Sciences*, pp. 168–172, Taipei, Taiwan, May 2011.

[28] D. T. Wang, F. Fu, X. Q. Rao, and Y. B. Ying, "Fruit traceability system based on processing and grading line," *Transactions of the Chinese Society of Agricultural Engineering*, vol. 25, pp. 228–236, 2013.

[29] P. L. Peng, C. Y. Liu, L. P. Li, L. Z. Ji, S. D. Y. Yuan, and C. C. Liu, "Multiscale multivariate fuzzy entropy analysis," *Acta Physica Sinica*, vol. 62, no. 12, https://doi.org/10.7498/aps.62.120512, Article ID 120512, 2013.

[30] M. U. Ahmed and D. P. Mandic, "Multivariate multiscale entropy analysis," *IEEE Signal Processing Letters*, vol. 19, no. 2, pp. 91–94, 2012.

[31] F. C. Morabito, D. Labate, F. L. Foresta, A. Bramanti, G. Morabito, and I. Palamara, "Multivariate multi-scale permutation entropy for complexity analysis of alzheimer's disease EEG," *Entropy*, vol. 14, no. 7, pp. 1186–1202, 2012.

[32] L. Cao, A. Mees, and K. Judd, "Dynamics from multivariate time series," *Physica D: Nonlinear Phenomena*, vol. 121, no. 1-2, pp. 75–88, 1998.

[33] G. N. Michael, "Benchmarking and stress testing a hadoop cluster with terasort," 2011, https://www.michael-noll.com/blog/2011/04/09/benchmarking-and-stress-testing-an-hadoop-clusterwith-terasort-testdfsio-nnbench-mrbench/.

[34] J. Mao, H. Yin, C. Cui, and T. L. Wang, "Based on Minitabconfirmedmost appropriate parameter of design of experiment," *Coal. Mine. Mach*, vol. 29, pp. 14–16, 2008.

[35] S. Keshani, W. R. W. Daud, M. M. Nourouzi, F. Namvar, M. Ghasemi, and M. Ghasemi, "Spray drying: an overview on wall deposition, process and modeling," *Journal of Food Engineering*, vol. 146, pp. 152–162, 2015.

[36] S. Kumar, R. Gokhale, and D. J. Burgess, "Quality by design approach to spray drying processing of crystalline nanosuspensions," *International Journal of Pharmaceutics*, vol. 464, no. 1-2, pp. 234–242, 2014.

[37] X. Q. Gong, C. Q. Jin, X. L. Wang, R. Zhang, and A. Y. Zhou, "Data-intensive science and engineering:requirements and challenges," *Chinese Journal of Computers*, vol. 35, no. 8, pp. 1563–1578, 2012, https://doi.org/10.3724/SP.J.1016.2012.01563.

[38] J. Q. Zou, Q. B. Zhou, P. Yang, W. Wu, and Q. Huang, "Integration and example analysis for farmland data management system of wireless sensor networks," *Transactions of the Chinese Society of Agricultural Engineering*, vol. 28, pp. 142–147, 2012.