


RESEARCH

Open Access



An efficient storage and service method for multi-source merging meteorological big data in cloud environment

Ming Yang^{1,3} , Wenchun He², Zhiqiang Zhang², Yongjun Xu², Heping Yang², Yufeng Chen¹ and Xiaolong Xu^{3,4,5,6*}

Abstract

With the development of the meteorological IoT (Internet of Things) and meteorological sensing network, the collected multi-source meteorological data have the characteristics of large amount of information, multidimensional and high accuracy. Cloud computing technology has been applied to the storage and service of meteorological big data. Although the constant evolution of big data storage technology is improving the storage and access of meteorological data, storage and service efficiency is still far from meeting multi-source big data requirements. Traditional methods have been used for the storage and service of meteorological data, and a number of problems still persist, such as a lack of unified storage structure, poor scalability, and poor service performance. In this study, an efficient storage and service method for multidimensional meteorological data is designed based on NoSQL big data storage technology and the multidimensional characteristics of meteorological data. In the process of data storage, multidimensional block compression technology and data structures are applied to store and transmit meteorological data. In service, heterogeneous NoSQL common components are designed to improve the heterogeneity of the NoSQL database. The results show that the proposed method has good storage transmission efficiency and versatility, and can effectively improve the efficiency of meteorological data storage and service in meteorological applications.

Keywords: Multi-source merging sensors data, Meteorological data storage, Meteorological data service, Distributed NoSQL, Semi/unstructured data

1 Introduction

The application and service of meteorology become increasingly common in human daily lives, and meteorological information has become crucial in various aspects of life [1]. Meteorological data collected mainly by meteorological IoT (Internet of Things) and meteorological sensor networks, including temperature, precipitation, pressure, and radar echo, are the basic data of meteorological application service. Meteorological sensor networks are data-centric networks, which process data for continuous meteorological data collected by sensors.

In addition, with the development of meteorological IoT and meteorological sensor network technology, especially the atmospheric remote sensing and IoT technology, the types and quantities of the collected meteorological data are increasing rapidly [2, 3]. Meanwhile, the meteorological data collected by meteorological IoT and meteorological sensor networks have the characteristics of big data. Meteorological data are mainly composed of a variety of data structures, including structured, semi-structured, and unstructured data [4]. Traditional relational database management systems (RDBMS) have been used for the analysis, processing, and storage requirements of structured data, such as automatic weather station data and station forecasting. The semi-structured and unstructured data are mainly stored in file management systems in various formats, such as NetCDF (Network Common Data Form) [5], GRIB (General Regularly-distributed Information in Binary form), BUFR (Binary Universal Form for the

*Correspondence: xlxu@nuist.edu.cn

³School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing, China

⁴Jiangsu Engineering Centre of Network Monitoring, Nanjing University of Information Science and Technology, Nanjing, China

Full list of author information is available at the end of the article

Representation of meteorological data) [6] and text formats. In addition, meteorological metadata [7] and multidimensional data index [8] are used for meteorological data storage and service.

However, the storage and service of a single traditional relational data and meteorological file can no longer meet the needs of high performance and high concurrency storage and service for big meteorological data. The traditional storage and service methods of meteorological data collected by meteorological sensors have been encountering many challenges in real-time big data storage and real-time processing and service. Thus, how to effectively implement the transmission, storage, and service of meteorological data has become an urgent issue.

Meteorological data not only need to be stored at high speed in high concurrency but also need to provide fast and efficient data product services. Although cloud computing and big data technologies have been applied to massive and complex meteorological datasets, the storage and service efficiency of meteorological data is still far from meeting the requirements. Therefore, it is necessary to design an efficient storage service method for meteorological data in cloud environment based on the concepts of NoSQL database systems such as Cassandra, MongoDB, HBase, Membase, and Redis.

In view of the above challenge, a high-dimensional meteorological data storage and service method with high generality, high storage efficiency, and high read-write and service performance are proposed in this paper to improve the scalability and storage service performance of big data storage for multidimensional meteorological data and build heterogeneous, scalable, and highly available application service frameworks based on multiple distributed NoSQL in cloud environments. Specifically, our main contributions are mainly in the following three aspects: Firstly, we provide a multidimensional meteorological data storage architecture in the distributed NoSQLs environments based on the design principle of distributed storage system for structured data (Bigtable) [9]. Then a block compression algorithm for multidimensional meteorological data, a time series extraction algorithm for any location, an adaptive

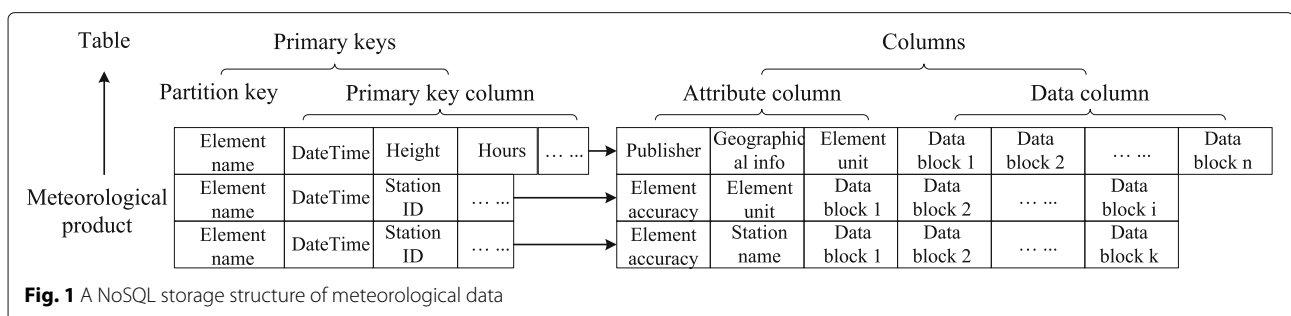
data time-to-live algorithm, and a unified meteorological data service method are designed. Finally, the experimental analysis is carried out to verify the performance of the method proposed in this paper.

The remainder of this paper is organized as follows. Section 2 describes the meteorological data storage structure applied to the multidimensional meteorological data. Section 3 presents the process methods, including the blocking algorithm, the interpolation algorithm of the geographic grid for the time series, the adaptive data time-to-live algorithm, and the unified heterogeneous service methods, used for the meteorological cloud applications. Section 4 describes the experimental environment and dataset used to evaluate and analyze the scalability, storage, and service performance of big data storage for multidimensional meteorological data. In Section 5, related works are summarized. Section 6 concludes the paper and presents the prospect for the future work.

2 Meteorological data storage structure

A large amount of meteorological data have nonstructural features, and unstructured data are mainly based on grid data or file data. Both types of data can be stored by multidimensional block compression method. NoSQL object data storage generally adopts column-oriented storage mode, and its storage structure ensures column scalability of data tables and high throughput of read/write I/O and is more suitable for field expansion characteristics and intensiveness of data tables in meteorological cloud data environments. The data analysis application avoids the maintenance pressure brought by the subsequent table structure changes and effectively improves the performance of the intensive data analysis. Based on the design principle of Google's BigTable [9] distributed data storage system, combined with the general characteristics of column storage of a distributed NoSQL database, a general NoSQL storage structure for meteorological data is designed.

As shown as Fig. 1, the NoSQL storage structure of meteorological data is mainly composed of tables, primary keys such as meteorological products and elements, and columns corresponding to meteorological data and



attributes. Among them, the meteorological element is used as a slice key (partition key) so that meteorological data are evenly distributed and stored in different storage nodes.

In the NoSQL storage structure of meteorological data, the primary key must be unique, mainly represented by the name, height, date, and time of meteorological elements. The attribute column is a data dynamic data column generated by the data blocking algorithm. The number of columns is dynamically generated according to the data. The attribute column also stores information from the meteorological data, such as geographical range, block algorithm identification, and compression mode.

According to different meteorological data, based on the general storage structure, different data storage organization methods are adopted. The commonly used meteorological data storage structure is as follows:

2.1 The storage structure of meteorological grid data

Meteorological grid data are widely used in meteorological field, such as numerical prediction products and radar grid products. These data have the characteristics of a large amount of information, multidimensionality, and high precision, and generally have three-dimensional features, as shown in Fig. 2.

Meteorological data require quick access to a region and single-point sequence and the ability to use meteorological algorithms for online distributed computing. The storage structure of meteorological grid data is mainly divided into three parts: primary key, attribute, and data, as shown in Fig. 3.

The primary key is mainly composed of meteorological elements, height, date, and time. The attributes mainly include information such as grid range, data attributes, and producer information.

2.2 The storage structure of meteorological file data

Meteorological file data are often applied to radar-based data files, satellite files, and so on. These data have the characteristics of strong continuity of data bytes and pertinence. Its storage structure is shown in Fig. 4.

The blocking algorithm mainly divides the data in bytes and the data are read in byte segments.

2.3 The storage structure of meteorological common data

A wider-range storage application is used to store meteorological data without a calculation function. It is mainly used as meteorological data for storing data carrier functions, such as picture data, radar data, satellite data, etc. Comparing grid and file data storage structure, its primary key, data, and attribute are more flexible. As shown in Fig. 5, the common data structure is represented by more general data volume, and the file data structure and grid structure are a subset of it.

3 Methods for meteorological data storage and service

In this section, the blocking and coding algorithm, the interpolation algorithm of the geographic grid for the time series, the adaptive data time-to-live algorithm, and the unified heterogeneous service method for multi-source merging meteorological big data in cloud environment are introduced.

3.1 The blocking and coding algorithm of multidimensional meteorological data

Semi-structured data are mainly grid data or raster data, which can be stored by block and compression. NoSQL object storage generally adopts column-oriented storage mode. Its storage structure guarantees the expansion of data table columns and the high throughput of I/O, avoids the maintenance pressure brought by the change of

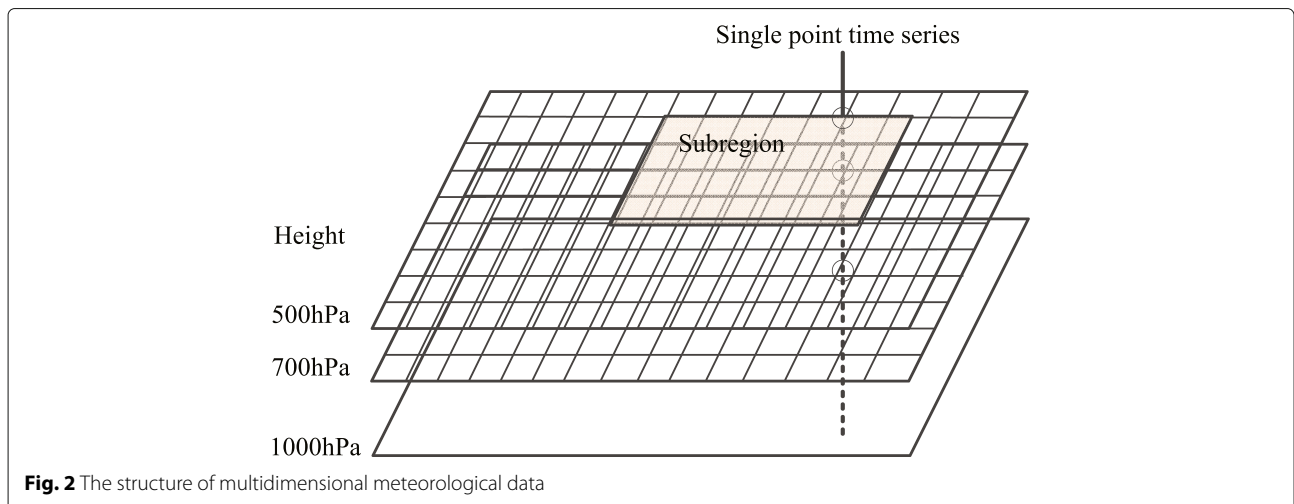


Fig. 2 The structure of multidimensional meteorological data

Primary key	Element name	Date time	Height	Hours
Attribute	Attribute information such as latitude and longitude range and data range of each dimension				
Data	Blocking compressed Meteorological grid data corresponding to the primary key				

Fig. 3 The storage structure of meteorological grid data

table structure, and effectively improves the throughput performance of data analysis.

Based on the characteristics of the NoSQL object storage and the actual characteristics of the meteorological grid data, the blocks are scaled according to the rank of grid data. The data of attribute columns are divided into rows and columns in a certain proportion. The rules of segmentation are as follows: the meteorological grid data is divided into $M \times N$ squares by the count of row and column. Suppose N is the count of cells in the vertical direction after segmentation, and M is the count of cells in the horizontal direction after segmentation, M and N are calculated by

$$\begin{cases} M = \frac{m}{x} + 1, & m \geq 0, x > 0 \\ N = \frac{n}{y} + 1, & n \geq 0, y > 0 \end{cases} \quad (1)$$

where m is the count of blocks in the custom vertical direction, n is the count of blocks in the custom horizontal direction, x is the count of cells in the vertical direction after segmentation, and y is the count of cells in the horizontal direction after segmentation.

The partitioned attribute column data is divided into $M \times N$ data columns and stored as one-dimensional arrays sequentially in the NoSQL database system. As shown in Fig. 6, each column is encoded in different bytes and compressed by GZip or LZ4.

According to the above method, when reading a range of data, the value of the attribute column can be read by calculating the column in which the data is located, without reading the whole row of grid field data row. Moreover, after data is partitioned and compressed, the amount of data read from the grid is greatly reduced.

The multidimensional meteorological grid data blocking algorithm helps to reduce the amount of query data

Primary key	Element name	Date time	File name
Attribute	Attribute information such as file type, size, creation time, etc.			
Data	Blocking compressed weather file data corresponding to the primary key			

Fig. 4 The storage structure of meteorological file data

Primary key	Element name	Reserved 1	Reserved 2
Attribute	Attribute information such as data description, size, creator, etc.			
Data	Blocked compressed weather binary data corresponding to the primary key			

Fig. 5 The storage structure of meteorological common data

and improve data access performance. For example, when a data query by the grid extraction algorithm intercepts a range-related block data set, single-point timing generation occurs. A single block column of multiple height field data is taken to form a time series.

3.2 The interpolation algorithm for generating the time series of a geographic grid

In meteorological applications, the history or future trends of a geographic location are often viewed, which requires retrieving a multidimensional time series of that location. Thus, suppose a time series set of data

$$T = \{V_{t1} + V_{t2} + \dots + V_{tn}\} \quad (2)$$

is given, where V is the coordinate value on a two-dimensional field at a certain time, t is the time vector, and T is the time series set of the multidimensional meteorological data.

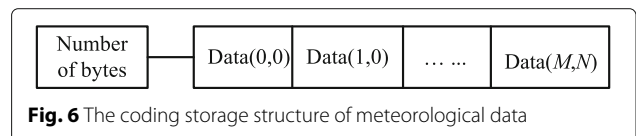
Bilinear interpolation is used to obtain the value of this point on a two-dimensional field of a layer. Mathematically, bilinear interpolation is a linear interpolation extension of an interpolation function with two variables. The core idea is to perform linear interpolation in two directions. First, the data block position is searched according to formulas (4) and (5). For the analysis and calculation, the total amount of block data is formulated as follows

$$Z_{m,n}(t) = \sum_{i=1}^m \sum_{j=1}^n \beta_{ij}(t) \quad (3)$$

where $Z_{m,n}(t)$ is the total amount of block data at time t , $\beta_{ij}(t)$ is the size of a single grid node data, m is the grid number of latitudes, and n is the grid number of longitudes.

The data block position is calculated by

$$\begin{cases} BI_x = (\lambda - \lambda_0) / \mu \\ BI_y = (\eta - \eta_0) / \mu \end{cases} \quad (4)$$



where BI_x and BI_y are the data block index in the latitude and longitude directions, respectively, μ is the size of a single block data, λ is the longitude value, λ_0 is the initial longitude value, η is the latitude value, and η_0 is the initial latitude value.

The data set in the block are obtained by formula (4), and then the index position in the block is calculated by formula (5).

$$\begin{cases} x = (\lambda - \lambda'_0)/dx \\ y = (\eta - \eta'_0)/dy \end{cases} \quad (5)$$

where x is the index value in longitude, y is the index value in latitude, dx is the longitude resolution, dy is the latitude resolution, λ'_0 is the initial longitude value in the data block, and η'_0 is the initial longitude value in the data block.

Then, the obtained data position index function in the data block is used to make a bilinear interpolation to point $p(x, y)$. Thus, it is assumed that we know the value of $f(p)$ at the four points $Q_{11} = (x_1, y_1)$, $Q_{12}(x_1, y_2)$, $Q_{21}(x_2, y_1)$, and $Q_{22}(x_2, y_2)$. The corresponding interpolation $f(x, y)$ is calculated by formula (6), formula (7), and formula (8). The linear interpolation in the x -direction is estimated by

$$f(x, y_1) \approx \frac{(x_2 - x)}{(x_2 - x_1)}f(Q_{11}) + \frac{(x - x_1)}{(x_2 - x_1)}f(Q_{21}) \quad (6)$$

$$f(x, y_2) \approx \frac{(x_2 - x)}{(x_2 - x_1)}f(Q_{12}) + \frac{(x - x_1)}{(x_2 - x_1)}f(Q_{22}) \quad (7)$$

The desired estimate is obtained by

$$f(p) \approx \frac{(y_2 - y)}{(y_2 - y_1)}f(x, y_1) + \frac{(y - y_1)}{(y_2 - y_1)}f(x, y_2) \quad (8)$$

Finally, according to the same steps, the data of the same position at other times are taken to form a time series of the multidimensional meteorological data.

3.3 Adaptive time-to-live algorithms

It is nothing but time on the period of time of data in meteorological NoSQL system that a unit of data can experience before it should be discarded. The TTL of meteorological NoSQL data reflects the whole lifetime of the meteorological data from input database to deletion. In order to improve the effect of time-to-live (TTL) in the meteorological NoSQL database, an adaptive algorithm for cleaning up expired meteorological data is implemented according to the frequency and time of data access. Suppose the calculation period of the time-to-live is $(0, \infty)$. At the current time t , the time-to-live of data could be calculated based on the frequency and time of data access in different time periods.

Some basic concepts are presented for adaptive time-to-live algorithms in the meteorological NoSQL system as follows.

Definition 1 (Data usage of E_t). *The data usage of E_t is defined by the amount of data access at time t , denoted as P_t .*

Definition 2 (Adaptive data expiration model). *The adaptive data expiration model can be described as a quaternion $\{P, T, \alpha, E\}$, where T is the data expiration time, P is the usage of data access and time, and $\alpha : P \times T \rightarrow E$ is the mapping determined by the data expiration algorithm, E is the set of outputs are represented.*

In the initial stage of data cleaning, the initial expiration time t_0 of design data is E_0 . With the increase of time, the expiration time decreases continuously and when equals to 0, the data will be automatically cleaned up by the system. In the process of time change, the data is accessed. With the increase of access frequency, the usage of the data increases, and the expiration time of the corresponding data will increase. Their relationship formulas are as follows:

$$E_t = \begin{cases} E_{\max} - t + P(t), E_{\max} \neq 0 \\ 1, E_{\max} = 0 \end{cases} \quad (9)$$

where E_t is the data live time at time t and when equals to 0, the data is deleted; E_{\max} is the maximum TTL time defined by the user and when equals to 0, the data will never expire; and $P(t)$ is the data usage at time t , which the relationship between data access frequency and time, calculated by formula (11).

The sigmoid function is used as the adaptive adjustment curve as shown in formula (10).

$$f(x) = \frac{1}{1 + \exp(-ax)} \quad (10)$$

The adaptive adjustment formula of the change rate of access frequency for solving the usability problem is designed by using sigmoid function as shown in formula (11).

$$P_t = \begin{cases} \frac{P_{\max} - P_{\min}}{1 + \exp\left(\frac{a(P_{t-1} - A_{\min t})}{A_{\text{avg}t} - A_{\min t}}\right)} + P_{\min}, P_{t-1} \geq P_{\min t} \\ P_{\min}, P_{t-1} < P_{\min t} \end{cases} \quad (11)$$

where $t = 1, 2, \dots, T$ is the current time, $T \in \infty$, a is the arbitrary constant, P_{\max} is the maximum usage of data, P_{\min} is the minimum usage of data, $A_{\min t}$ is the minimum access from 0 to t , $A_{\text{avg}t}$ is the average access from 0 to t , P_{t-1} the data usage of the previous period, and P_t the data usage of the current period t .

As shown in formulas (10) and (11), the change rate of usage is adjusted nonlinearly with sigmoid curve between the minimum and average data access frequencies between two adjacent periods. Obviously, the distribution of usage is improved when most of the time

periods have similar access frequencies and the minimum data access frequencies are close to the average access frequencies.

The process of adaptive adjustment with data access frequencies is shown in Fig. 7. When the accessibility varies with the access frequency of the current period, the accessibility value is obtained, so that the data expiration time can be adjusted adaptively. When the data expiration time is 0, the data will be automatically cleared. The adaptive adjustment curve of radius change rate should be changed slowly everywhere, so that it can be improved in a large area. Secondly, when the difference is large, it ensures that the adaptive adjustment curve does not tend to be linear, which indicates that high access frequency does not have normality. Finally, it ensures a certain degree of distribution in accessibility. At the same time, in order to retain better distribution as much as possible, the adaptive adjustment curve at a smoother position should be adopted.

Therefore, in order to automatically adjust the expiration time of the data and improve the efficiency of data use, the relationship between expiration time and access frequency at each time is designed in this paper.

3.4 Unified heterogeneous service architecture and method

To improve the service capability of data storage and shield the difference of the underlying NoSQL data system, a unified heterogeneous service method for meteorological data was studied. It uses a three-layer abstract structure, as shown in Fig. 8.

The NoSQL database service layer is based on a variety of NoSQL databases and provides a variety of NoSQL database read-write interfaces and other general operation interfaces and implementation classes, automatically adapting operations for different database characteristics. Shielding the underlying NoSQL data storage platform from the application, it automatically adapts to all types of NoSQL data storage platforms, providing very high service heterogeneity.

The service implementation layer mainly implements the core operations of the unified heterogeneous services of meteorological data, including table spaces, tables, data, and related meteorological algorithms. It organizes data in the form of table spaces, tables, and columns and dynamically and seamlessly expands meteorological data by segmenting meteorological data. It provides common meteorological data operations, supports meteorological grid classes, file classes, site class data read-write spatial analysis, etc., providing high consistency. The service implementation layer also provides algorithms such as adaptive data segmentation, compression, mesh extraction, interpolation, and single-point timing to achieve meteorological data processing.

The service interface layer provides data unified access interface service and data operation interfaces, including data unified access interface, database connection interface, table space interface, table interface, and meteorological data algorithm interface. It can be combined with commonly used distributed service components to provide distributed component-type meteorological data

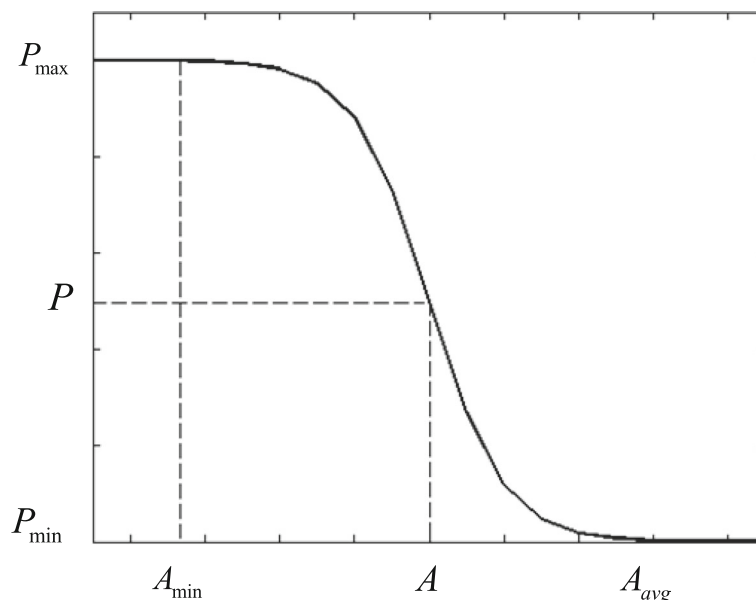


Fig. 7 Adaptive adjustment curve

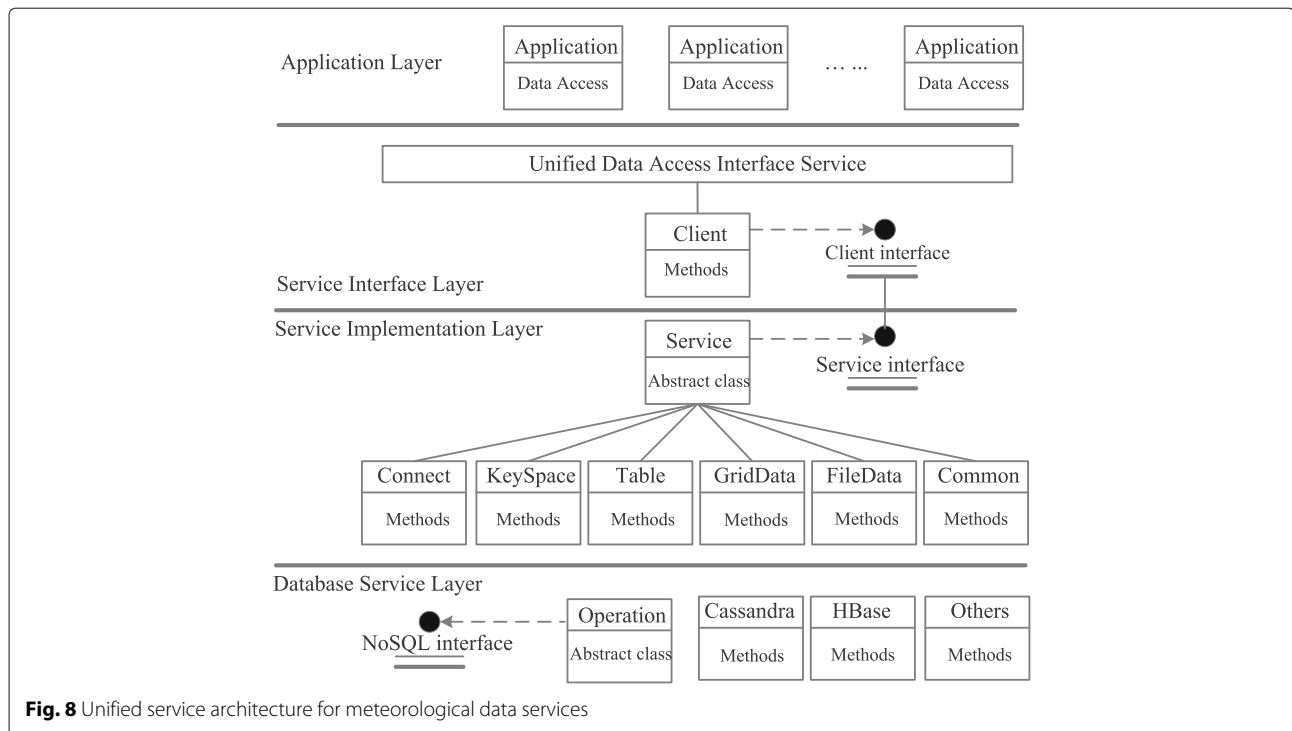


Fig. 8 Unified service architecture for meteorological data services

interface services. Applications can access meteorological data by RESTful Interface, Web Service Interface, and distributed Client SDK, which uses meteorological data algorithm as component or service of application program.

Meteorological Data Unified Heterogeneous Services data are stored on the NoSQL data storage platform and have multiple backups, providing fast access performance and extremely high data reliability.

4 Experimental and analysis

Using the meteorological private cloud experimental project, the research and application of a multidimensional meteorological data cloud storage and service based on a private cloud environment was carried out. In the exploration and application, the test environment, test data, test results, analysis, and application examples were carried out. The results show that the multidimensional meteorological data cloud storage service method introduced in the article has achieved good results.

4.1 Experimental context

The experimental environment is based on the meteorological private cloud platform. The experimental environment consists of five cloud servers, two distributed relational databases, three NoSQL Object Table Storage Systems, two NoSQL File Storage Systems, and three clients. The detailed configuration is shown in Table 1.

4.2 Experimental data set

The experimental data are commonly used in meteorological services, such as Numerical Forecast Products (NFP), Radar Precipitation Forecasting (QPF), Meteorological Grid Product (MGP), FY4A Sate Product (FSP), and Radar Echo Product (REP), which have the typical characteristics of large data volume, high precision, and multi-dimensional. The data types are shown in Table 2. Through the data collection and storage subsystem of the meteorological cloud service system, different cloud data storage methods and nodes are adopted according to different data types. Then, according to different application requirements, the basic data are processed through the processing subsystem of the meteorological cloud service system and then into different data products before being written back to the data collection storage subsystem.

Table 1 The parameters of cloud environment

Terms	Number	CPU (core)/memory (GB)/data disk (GB)
Distributed Relational Database	2	32/128/3072
NoSQL Object Table Storage System	3	48/192/5120
NoSQL File Storage System	2	32/128/10240
Test Client Server	3	24/96/900
Distributed Service Server	3	48/192/1024
Data Interface Service Server	2	32/192/600

Table 2 The set of experimental data

Data name	Storage type	Storage service
NFP	Grid data	NoSQL Object Table Storage System
QPF	Grid data	NoSQL Object Table Storage System
MGP	Grid data	NoSQL Object Table Storage System
FSP	Raster data	NoSQL Object Table Storage System
REP	File data	NoSQL File Storage System

Finally, data services are provided for different service systems through the system's data sharing service.

4.3 Experimental analysis

The method proposed in this paper tends to store and apply meteorological unstructured data more efficiently, which is relevant to data storage structure, retrieval method, and service efficiency of unstructured meteorological data. Therefore, the application experiments are constructed by meteorological private cloud experimental environment on storage performance and service performance. The data used in the experiment are usually high precision and high volume data.

4.3.1 Evaluation of the storage performance

Storage performance of meteorological data is a key factor to determine the performance of meteorological services, thus we evaluate on this value to discuss the storage performance achieved by REP, QPF, and FSP with different datasets.

Meteorological data storage performance is mainly analyzed in terms of storage time and data storage capacity. Thus, the multidimensional meteorological data block and coding algorithm introduced in this paper is used to store semi/unstructured meteorological data. The commonly used meteorological data type comparison analysis test and the storage average performance are listed in Table 3.

As shown in Table 3, the average storage time of the three representative data types is less than 100 ms. In addition, because the data stored in meteorological private cloud test environment adopts the binary digit and compression coding introduced in this paper, the amount of data stored is lower than that of the original file. Therefore, reducing the amount of data stored leads to

Table 3 Comparison of the storage results for meteorological data

Sample data	Average storage time (ms)	Original file size (MB)	New storage data size (MB)
REP	68.3	2.6	0.15
QPF	91.3	5.8	0.22
FSP	67.0	2.5	0.13

more efficient storage and data services than the original data.

The new NoSQL meteorological data storage methods introduced in this paper by REP, QPF, and FSP with different datasets are 17.3, 25.9, and 19.2 times less than the original file storage methods, respectively.

4.3.2 Evaluation of the service performance

In order to validate the proposed meteorological data service method, the query time of data API on the meteorological cloud application using NFP, MGP, and FSP using three different types of datasets are listed in Table 4, including any latitude and longitude grid time sequences data, hourly automatic station precipitation grid data, and satellite raster data.

The main purpose of the experiment is to show that the storage method of meteorological data and the uniform heterogeneous service method of meteorological data are introduced to test the response performance of application services and can be applied to realistic cases. Therefore, we evaluate on this value to discuss the service performance achieved by NFP, MGP, and FSP with different datasets. The service performance is referenced to the response time of requests for data services in different data service modes in the cloud environment.

The results on the experimental dataset using the proposed methods with three sample data are illustrated in Figs. 9, 10, and 11, respectively.

As illustrated in Fig. 9, the average response time of any longitude and latitude time series of NFP is 178.3 ms, the minimum response time is 153 ms, and the maximum response time is 193 ms, which can well meet the application needs of the actual cases.

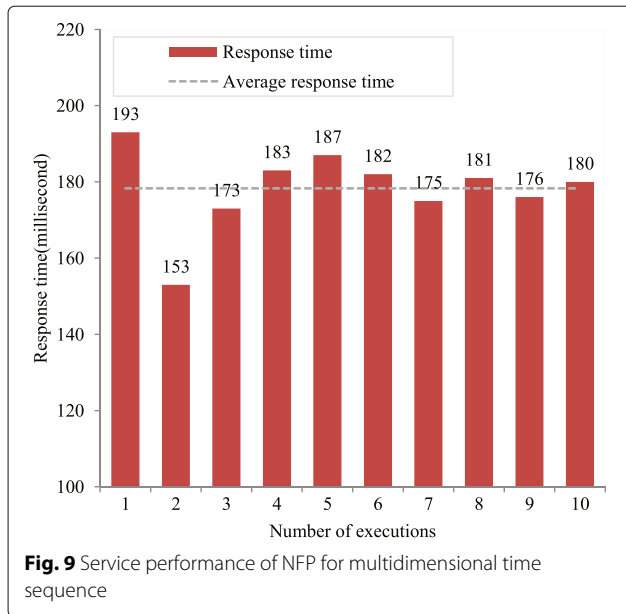
It is intuitive from Fig. 9 that the grid time series of any longitude and latitude of NFP can be obtained quickly in at millisecond level, since the time series linear method introduced in this paper is used as the method of calculating longitude and latitude data of each grid field.

From Fig. 10, it can be seen intuitively that the average response time of MGP is 67.2 ms, the minimum response time is 52 ms, and the maximum response time is 112 ms, which can well meet the application needs of the actual cases.

MGP grid data can be acquired quickly at millisecond level, because the block method introduced in this paper is adopted to acquire data blocks in the grid by different

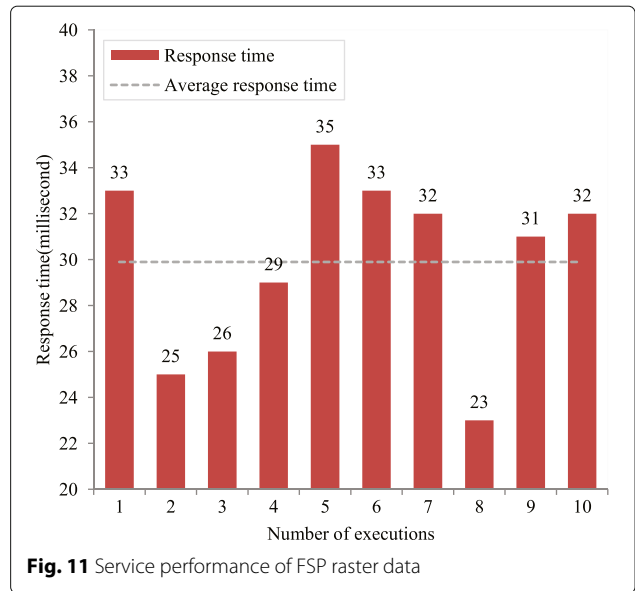
Table 4 Evaluation method of the service performance

Data name	Service mode	Request type
NFP	Time sequence data	Data API
MGP	Grid data	Data API
FS	Raster data	Data API



grid ranges, which could reduce the amount of data and transmission time.

The evaluation of the average response time is also conducted by FSP using raster data. Figure 11 shows the average response time of FSP raster product is 29.9 ms, where we can find out that the proposed meteorological unstructured data storage and service methods are also effective for raster data. The main reason is that the raster data is stored in a multi-level pyramid mode and read in a certain geographical range, so the amount of raster data read by application is smaller and faster.



4.4 Discussion

Based on the above experiments, it can be concluded that the meteorological unstructured data storage and service methods proposed in this paper can produce good performance due to its special feature of data storage and service.

The experimental results on storage performance and service performance of meteorological data show that it is effective and efficient to store and apply meteorological semi/unstructured data using the proposed methods in this paper and can meet the needs of meteorological services.

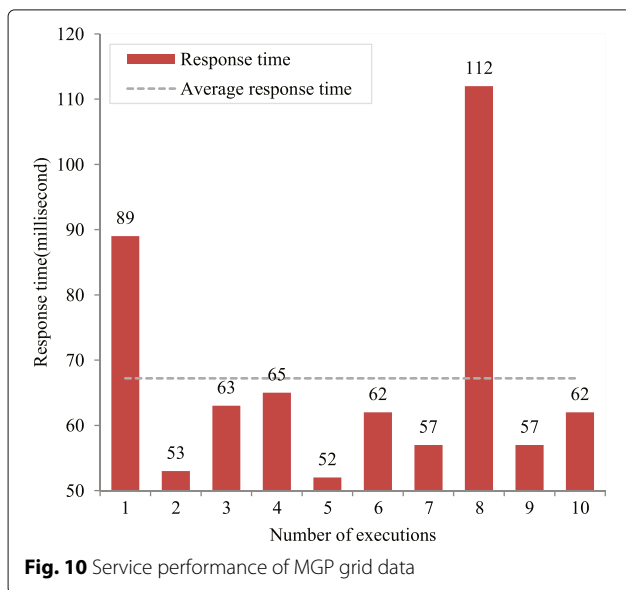
5 Application example

The method described in the article provides an efficient data storage and unified service interface for the Zhejiang Meteorological Typhoon Service Website. Through the Zhejiang Meteorological Typhoon Service Website deployed on the cloud platform, the test application has achieved certain results in terms of data storage access performance. The interface application effect is shown in Fig. 12. In the Zhejiang Meteorological Typhoon Service Website, the Meteorological Cloud Service System provides distributed data services and components.

As shown in Fig. 12 is the application effect of hourly rainfall forecast and cumulative rainfall forecast in grid time series of arbitrary longitude and latitude. The average access time of arbitrary longitude and latitude reaches 1.16 ms in a certain network environment, which satisfies application needs well.

6 Related work

In the meteorological big data era, meteorological data collected mainly by meteorological IoT and meteorological sensor networks play a fundamental role in processes



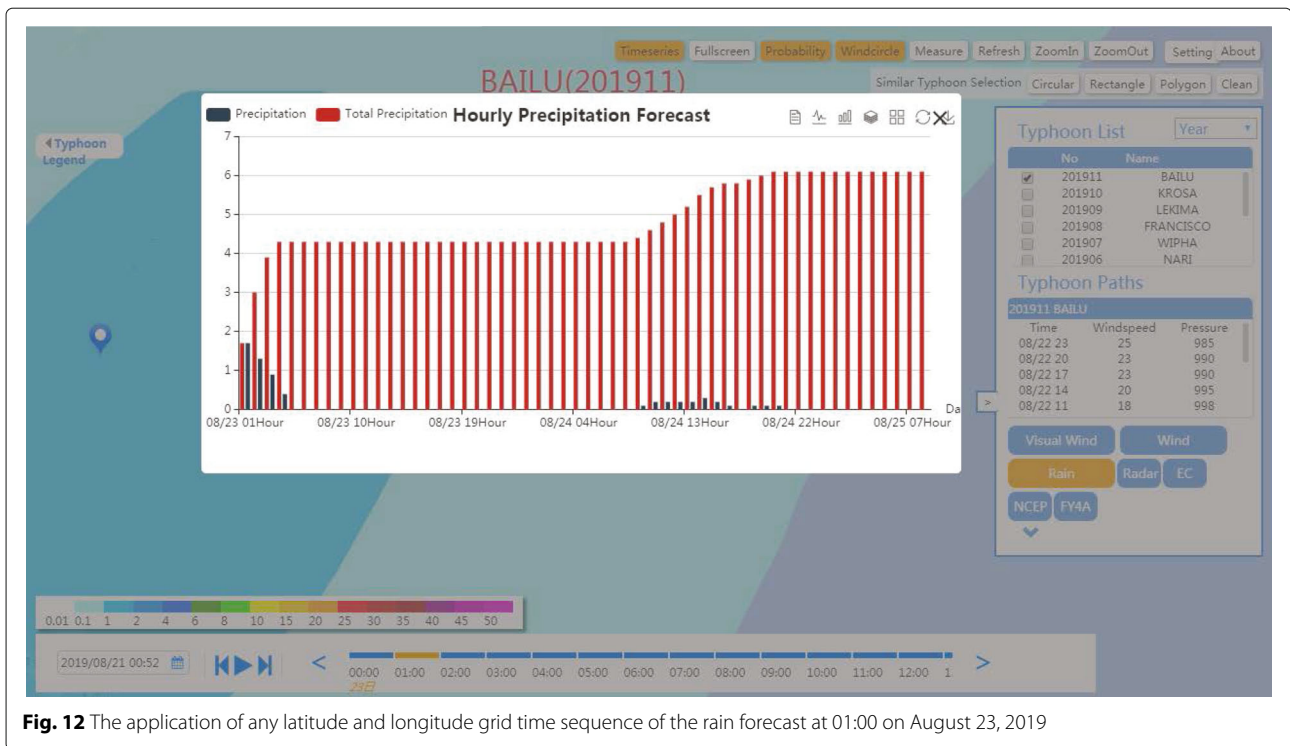


Fig. 12 The application of any latitude and longitude grid time sequence of the rain forecast at 01:00 on August 23, 2019

of meteorological services. However, the traditional storage and service methods of meteorological data are difficult to provide efficient storage and service in real-time big data storage, real-time processing and service, etc. The collecting, processing, and storage methods of meteorological data are an effective factor to promote the efficiency of meteorological data.

There is currently a large amount that has been investigated to realize the storage methods for data management in many fields. In [4], Yang et al. analyzed the meteorological data storage and processing method based on the structural and semi/unstructured characteristics of meteorological data. Kim et al. [5] designed a high-performance input/output (I/O) library for the Korean Integrated Model, which is a consistent and efficient approach for input and output of essential data in this particular grid structure in a multiprocessing environment. Wan and Zhao et al. [8] presented an energy- and time-efficient multidimensional data indexing scheme, which is designed to answer range query. Chang et al. [9] described the simple data model provided by Bigtable, which gives clients dynamic control over data layout and format, and we describe the design and implementation of Bigtable. In other aspects, many methods for the big data have been proposed to ensure the effectiveness of the data [10–12]. In [13], Ding et al. proposed a long video caption generation algorithm for big video data retrieval.

At the moment, with the development of IoT, sensor networks, and cloud computing, they have become

common technologies in data management. Xu et al. [14] designed a computation offloading method over big data for IoT-enabled cloud-edge computing and an IoT-oriented data placement method with privacy preservation in cloud environment [15]. Furthermore, the cloud-edge computing and 5G network are expected to play a more important part in the next generation meteorological data acquisition and processing [16–18], to name a few. In [19], Wang et al. presented a tensor-based cloud-edge computing framework that mainly includes the cloud and edge planes. In [20], a blockchain-enabled computation offloading method was proposed.

However, the service structure and method of data services in cloud environment can improve service performance and response speed. Xiong et al. [2] designed the principle, system structure and data flow, and core technologies of China Integrated Meteorological Information Sharing System (CIMISS). In [21], Liu et al. designed a blockchain-based framework for data integrity service base on cloud IoT applications. In [22], Zaman et al. proposed a resource allocation framework of interconnected edge cloud data centers using software-defined networking. Bhattacharya et al. [23] proposed a framework of metrics which we used to conduct an in-depth performance and cost benefit analysis of two standard Hadoop infrastructural choices, i.e., a platform as a service (PaaS) on-demand cloud setup and a local organizational setup. Mohamed et al. [24] provided insights and knowledge on the existing big data platforms and their application

domains, the advantages and disadvantages of big data tools, big data analytics techniques and their use, and new research opportunities in future development of big data systems.

In addition, there are also some research related to the frontier technologies [25–28] and other related methods [29–31] [32], such as fog computing, virtual machines, and mobile edge computing, which will contribute to the efficiency of future meteorological data processing.

As far as we know, there are few studies on meteorological big data storage and service methods in distributed cloud environment which aims to realize the effectively stored and quickly served meteorological data in cloud environment.

7 Conclusion and future work

The meteorological big data storage and service methods proposed in this paper in cloud environment, combined with the multidimensional characteristics of meteorological data, are used to design a meteorological data storage structure, storage method, and unified heterogeneous service method based on the NoSQL storage system in meteorological private cloud experimental environment. The research results show that the methods described in this paper can effectively store and quickly serve the meteorological data, which effectively improve the performance of meteorological data storage and service:

(1) Based on the distributed NoSQL big data storage database, storing multidimensional meteorological data can effectively improve its storage efficiency of meteorological data.

(2) In the process of data storage, the meteorological data block algorithm and time series interpolation algorithm are used to realize the meteorological data transmission, which has good data retrieval and transmission efficiency and can meet the needs of large-scale meteorological data in meteorological applications.

(3) The unified heterogeneous service method of meteorological data can improve the service capability of data storage, shield the difference of the underlying NoSQL data system, and reduce the cost of learning NoSQL data system for developers.

With the continuous development of distributed NoSQL big data application technology, the meteorological data storage and service method and its application mode will play an important role in the modernization and integration of meteorological services.

For future work, we try to analyze and design an index-based distributed data storage and coding method for meteorological data in cloud environment. The idea is that the encoded large data are stored on different nodes by indexing the location of the data. Meanwhile, it is conducive to faster access and larger data storage.

Furthermore, the work will explore more characteristics of the edge computing and apply them into the big meteorological data storage and service to enhance the performance.

Abbreviations

BUFR: Binary Universal Form for the Representation of meteorological data; GRIB: General Regularly-distributed Information in Binary form; IoT: Internet of Things; NetCDF: Network Common Data Form

Acknowledgements

This work is supported by the Major Project of Zhejiang Province fund and the Key Project of Zhejiang Meteorological Service fund.

This research is supported by the Major Project of Zhejiang Province under grant no. 2017C03035. Besides, this work is also supported by the Key Project of Zhejiang Meteorological Service under grant no. 2017ZD17 and no. 2018ZD03.

Authors' contributions

MY, XX, WH, ZZ, YX, HY, and YC conceived and designed the study. MY and YX performed the simulations. MY and XX wrote the paper. All authors reviewed and edited the manuscript. All authors read and approved the final manuscript.

Availability of data and materials

In addition to radar and satellite data that are not available for confidential reasons, data sets supporting the conclusions of this article is available which can be downloaded at <https://pan.baidu.com/s/184JDZDs8JqrP9JeeF07jKg>. password: 94gp.

Competing interests

The authors declare that they have no competing interests.

Author details

¹Zhejiang Meteorological Information Network Center, Hangzhou, China. ²National Meteorological Information Center, Beijing, China. ³School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing, China. ⁴Jiangsu Engineering Centre of Network Monitoring, Nanjing University of Information Science and Technology, Nanjing, China. ⁵State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China. ⁶Department of Computer Science and Engineering, Michigan State University, East Lansing, MI, USA.

Received: 1 July 2019 Accepted: 10 October 2019

Published online: 29 October 2019

References

1. H. T. Reda, P. T. Daely, J. Jeevan Kharel, S. Y. Shin, On the application of IoT: Meteorological information display system based on LoRa wireless communication. *Iete Tech. Rev.* **35**(3), 1–10 (2017)
2. A. Xiong, Z. Fang, W. Ying, X. Zhang, G. Feng, D. Li, X. Tan, M. Qiang, Design and implementation of China integrated meteorological information sharing system (CIMISS). *J. Appl. Meteorol. Sci.* **26**(4), 500–512 (2015)
3. Y. Ji, C. Sun, Y. Liu, A method for optimizing storage efficiency of meteorological data in CIMISS. *Meteorol. Sci. Technol.* **45**(1), 30–35 (2017)
4. M. Yang, Y. Chen, Q. Chen, X. Yun, Z. Gao, C. You, Exploration and application of meteorological data storage method based on cloud data storage. *Meteorol. Sci. Technol.* **45**(6), 1017–1021 (2017)
5. J. Kim, Y. C. Kwon, T. H. Kim, A scalable high-performance I/O system for a numerical weather forecast model on the cubed-sphere grid. *Asia. Pac. J. Atmos. Sci.* **54**(S1), 403–412 (2018)
6. J. Caron, J. Oxelson, BUFR and GRIB file formats considered harmful for data archiving. *Egu General Assembly* (2013)
7. T. Zhu, Development of DAR Metadata for Meteorological Data in WIS. *J. Appl. Meteorol. Sci.* **23**(2), 238–244 (2012)
8. S. Wan, Y. Zhao, T. Wang, Z. Gu, Q. H. Abbasi, K.-K. R. Choo, Multi-dimensional data indexing and range query processing via Voronoi diagram for Internet of Things. *Futur. Gener. Comput. Syst.* **91**, 382–391 (2019)
9. F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, R. E. Gruber, Bigtable: A distributed storage system for structured data. *ACM Trans. Comput. Syst.* **26**(2), 1–26 (2008)

10. X. Wang, W. Wang, L. T. Yang, S. Liao, D. Yin, M. J. Deen, A distributed HOSVD method with its incremental computation for big data in cyber-physical-social systems. *IEEE Trans. Comput. Soc. Syst.* **5**(2), 481–492 (2018)
11. X. Xu, Q. Liu, Y. Luo, K. Peng, X. Zhang, S. Meng, L. Qi, A computation offloading method over big data for IoT-enabled cloud-edge computing. *Futur. Gener. Comput. Syst.* **95**, 522–533 (2019)
12. X. Wang, L. T. Yang, L. Kuang, X. Liu, Q. Zhang, M. J. Deen, A tensor-based big-data-driven routing recommendation approach for heterogeneous networks. *IEEE Netw.* **33**(1), 64–69 (2019)
13. S. Ding, S. Qu, Y. Xi, S. Wan, A long video caption generation algorithm for big video data retrieval. *Futur. Gener. Comput. Syst.* **93**, 583–595 (2019)
14. X. Xu, Y. Xue, L. Qi, Y. Yuan, X. Zhang, T. Umer, S. Wan, An edge computing-enabled computation offloading method with privacy preservation for internet of connected vehicles. *Futur. Gener. Comput. Syst.* **96**, 89–100 (2019)
15. X. Xu, S. Fu, L. Qi, X. Zhang, Q. Liu, Q. He, S. Li, An IoT-oriented data placement method with privacy preservation in cloud environment. *J. Netw. Comput. Appl.* **124**, 148–157 (2018)
16. X. Xu, Y. Li, T. Huang, Y. Xue, K. Peng, L. Qi, W. Dou, An energy-aware computation offloading method for smart edge computing in wireless metropolitan area networks. *J. Netw. Comput. Appl.* **133**, 75–85 (2019)
17. L. Ren, X. Cheng, X. Wang, J. Cui, L. Zhang, Multi-scale dense gate recurrent unit networks for bearing remaining useful life prediction. *Futur. Gener. Comput. Syst.* **94**, 601–609 (2018)
18. X. Xu, X. Liu, L. Qi, Y. Chen, Z. Ding, J. Shi, Energy-efficient virtual machine scheduling across cloudlets in wireless metropolitan area networks. *Mob. Netw. Appl.*, 1–15 (2019). <https://doi.org/10.1007/s11036-019-01242-6>
19. X. Wang, L. T. Yang, X. Xia, J. Jin, M. J. Deen, A cloud-edge computing framework for cyber-physical-social services. *IEEE Commun. Mag.* **55**(11), 80–85 (2017)
20. X. Xu, X. Zhang, G. Gao, Y. Xue, L. Qi, W. Dou, BeCome: Blockchain-enabled computation offloading for IoT in mobile edge computing. *IEEE Trans. Ind. Inform.*, 1–1 (2019)
21. B. Liu, X. L. Yu, S. Chen, X. Xu, L. Zhu, in *2017 IEEE International Conference on Web Services (ICWS)*. Blockchain based data integrity service framework for IoT data (IEEE, 2017). <https://doi.org/10.1109/icws.2017.54>
22. F. A. Zaman, A. Jarray, A. Karmouch, Software defined network-based edge cloud resource allocation framework. *IEEE Access.* **7**, 10672–10690 (2019)
23. D. Bhattacharya, F. Currim, S. Ram, Evaluating distributed computing infrastructures: an empirical study comparing Hadoop deployments on cloud and local systems. *IEEE Trans. Cloud Comput.*, 1–1 (2019). <https://doi.org/10.1109/tcc.2019.2902377>
24. A. Mohamed, M. K. Najafabadi, Y. B. Wah, E. A. K. Zaman, R. Maskat, The state of the art and taxonomy of big data analytics: view from new big data framework. *Artif. Intell. Rev.* **3**, 1–49 (2019)
25. J. Liu, W. Wang, D. Li, S. Wan, H. Liu, Role of gifts in decision making: an endowment effect incentive mechanism for offloading in the IoV. *IEEE Int. Things J.* **6**(4), 6933–6951 (2019)
26. X. Xu, S. Fu, Q. Cai, W. Tian, W. Liu, W. Dou, X. Sun, A. X. Liu, Dynamic resource allocation for load balancing in fog environment. *Wirel. Commun. Mob. Comput.*, 2018 (2018)
27. W. Li, X. Liu, J. Liu, P. Chen, S. Wan, X. Cui, On improving the accuracy with auto-encoder on conjunctivitis. *Appl. Soft Comput.* **81**, 105489 (2019)
28. L. Wang, H. Zhen, X. Fang, S. Wan, W. Ding, Y. Guo, A unified two-parallel-branch deep neural network for joint gland contour and segmentation learning. *Futur. Gener. Comput. Syst.* **100**, 316–324 (2019)
29. Z. Gao, H.-Z. Xuan, H. Zhang, S. Wan, K.-K. R. Choo, Adaptive fusion and category-level dictionary learning model for multi-view human action recognition. *IEEE Int. Things J.*, 1–1 (2019). <https://doi.org/10.1109/jiot.2019.2911669>
30. Z. Gao, D.Y. Wang, S. H. Wan, H. Zhang, Y. L. Wang, Cognitive-inspired class-statistic matching with triple-constrain for camera free 3d object retrieval. *Futur. Gener. Comput. Syst.* **94**, 641–653 (2019)
31. X. Xu, R. Huang, R. Dou, Y. Li, J. Zhang, T. Huang, W. Yu, Energy-efficient cloudlet management for privacy preservation in wireless metropolitan area networks. *Secur. Commun. Netw.* **2018**, 1–13 (2018). <https://doi.org/10.1155/2018/8180451>
32. X. Wang, L. T. Yang, X. Chen, M. J. Deen, J. Jin, Improved multi-order distributed HOSVD with its incremental computing for smart city services.

IEEE Trans. Sustain. Comput., 1–1 (2018). <https://doi.org/10.1109/tsusc.2018.2881439>

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Submit your manuscript to a SpringerOpen[®] journal and benefit from:

- Convenient online submission
- Rigorous peer review
- Open access: articles freely available online
- High visibility within the field
- Retaining the copyright to your article

Submit your next manuscript at ► [springeropen.com](https://www.springeropen.com)
