Li et al. EURASIP Journal on Wireless Communications and Networking (2017) 2017:45
DOI 10.1186/s13638-017-0827-1

EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

A novel HBase data storage in wireless sensor networks

Xiang Li[1], Zhuo Li[2,3,4*], Xirong Ma[1] and Can Liu[1]

Abstract

Based on the characteristics of data storage structure of HBase, and under the guidance of the storage optimization strategy, an efficient real-time storage model is designed. The real-time flow of data, which has fast storage in the cluster database, will satisfy the user's multiple needs of the data storage performance. Extenics primitives are used to integrate heterogeneous data-sets stored in the HBase database, cross-regional wireless sensor network data, and global data storage management catalogs for a double layer distributed storage architecture to improve storage and access efficiency. Under the circumstances that the storage space is insufficient or the storage space of the cluster text system is too massive, the HBase database cluster dynamic updating and optimizing the database space are also the research direction of this paper. Finally, the article carries on the experiment and concludes the appraisal through simulation.

Keywords: Wireless sensor network, HBase, Cluster, Real-time storage, Hadoop

1 Introduction

With the development of Internet of Things in China, there are problems of a large amount of information in information technology, and as the information storage of the end Internet of Things, wireless sensor network spreads more widely in a larger scale and sends greater amount of information [1, 2]. The accumulation of time and the increasing amount of data storage will lead to low database performance, and the system cannot be within a reasonable time to meet the needs of the users. Therefore, according to the actual needs of the database, choosing the appropriate way to improve storage for the users and retrieval efficiency becomes more important. With the existing database based on relationship and object model, the storage of complicated data can be solved, but the cost of storage and system consumption are large [3, 4]. Wireless sensor network has high real-time requirements for data analysis, but in practical application, many deployment wireless sensor networks will lead to the current processing of sensor data [5, 6]. Therefore, the regional storage cluster needs to be built. Every region has its own storage cluster of data, and the global data memory should be built to save the information of every region.

In this paper, under the guidance of the storage optimization strategy, we design an efficient real-time storage model—the real-time flow of data, which will be quickly stored in the cluster database to satisfy the multiple needs of the users who require data storage performance. In addition, the historical stream data stored in the text form will be migrated to the HBase cluster database in high efficiency and high stability. When the storage space is insufficient or the storage space of the text system of the cluster is too massive, efficient and simple dynamic updating of the HBase database cluster will help optimize the database space usage and it is discussed in this paper.

2 Wireless sensor networks

The data storage of wireless sensor network is divided into internal data storage and external application data storage, and the wireless network is made by sensors and sink nodes [7, 8]. Through the wireless network communication, a large number of sensor nodes will be deployed within or near a monitoring area and a network can be constituted by self-organization [9, 10]. Wireless sensor network accumulates data collected by sensors through gathering nodes, which will realize the interaction between data and application through gateway [11, 12]. At the same time, wireless sensor network

* Correspondence: zhuoli@cs.arizona.edu
[2]College of Electronic and Communication Engineering, Tianjin Normal University, 300087 Tianjin, China
[3]Department of Computer Science, The University of Arizona, Tucson, Pima County, AZ 85705, USA
Full list of author information is available at the end of the article

has a scale and its deployment is intensive. Thousands of sensor nodes are deployed within a wider monitoring area, so as to get a large amount of information, and thousands of sensor nodes are deployed within a narrower monitoring area, so as to get accurate data.

3 Storage solutions

Extenics argues that any material things and relationships in the objective world can be expressed by primitive formalization. Computers can get smarter after dealing with the contradiction between primitive extenics, and solving primitive data storage is of top priority to realize intelligence.

3.1 Distributed caching system

The cache entry of the distributed cache system exists in the form of <key, value>. Its structure is shown in Fig. 1. It uses the caching technique of memory cache query in order to temporarily store the results obtained from the database memory for the first time.

When the user requests the same memory, it can directly read the database for users to re-use, both reducing the request time and enhancing the efficiency of the system.

3.2 Distributed file system

Hadoop distributed file system is suitable for large-scale data sets, it can provide high-throughput data access and high fault tolerance, and it can be deployed in low-cost hardware features. Hadoop is a master-slave structure, comprised of the structure of the management system by the namespace, file-to-file mapping, and file-to-data node mapping. This three-part metadata information consists of a name node and a data node responsible for reading and writing data blocks. This structure is shown in Fig. 2.

3.3 Primitive HBase database storage

Extenics argues that any material things and relationships in the objective world can be expressed by using primitive formalization. Computers can get smarter after dealing with the contradiction between primitive extenics, and solving primitive data storage is of top priority to realize intelligence.

Definition: set $U = \{U_1, U_2 \cdots U_n\}$ is defined as the research object collection (including the collection of materials, things, and relationship), set $O = \{O_1, O_2 \cdots O_m\}$ is defined as the characteristic collection of research object, the value is Va_{ij} of object U_i about characteristic O_j, in which the value range of $i\ j$ is $i = 1, 2, 3 \cdots n$; $j = 1, 2, 3 \cdots m$, so Va_{ij} is called i as primitive and j as dimension primitive. When the value of Va_{ij} is empty, the object U_i has no data relationship of corresponding characteristic O_j.

HBase is a storage system which is distributed and column-oriented and is suitable for unstructured data storage and can provide algorithm of real-time writing, reading, and random access [13, 14]. The storage system of HBase is column-oriented in table form, and it aromatically divides the tables into rows and columns. Every region contains a subset of all rows of tables, which is identified by Timestamp. The master nodes are responsible for guiding and coordinating multiple region servers and responding to the clients' requests for reading and writing.

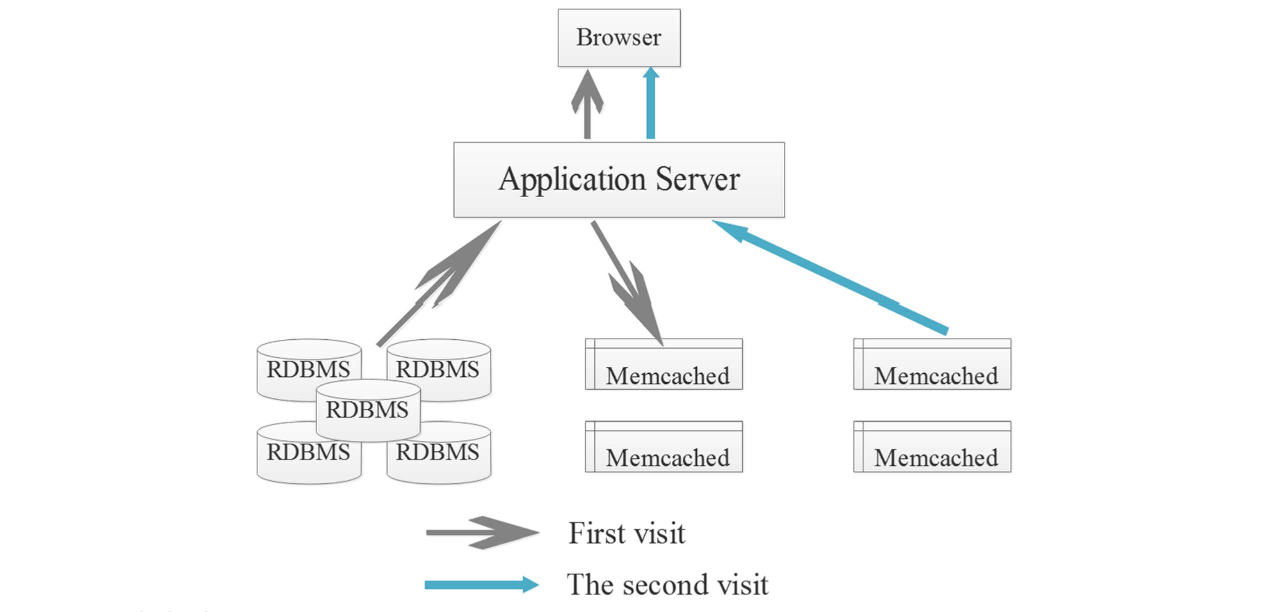

Fig. 1 Memcached architecture

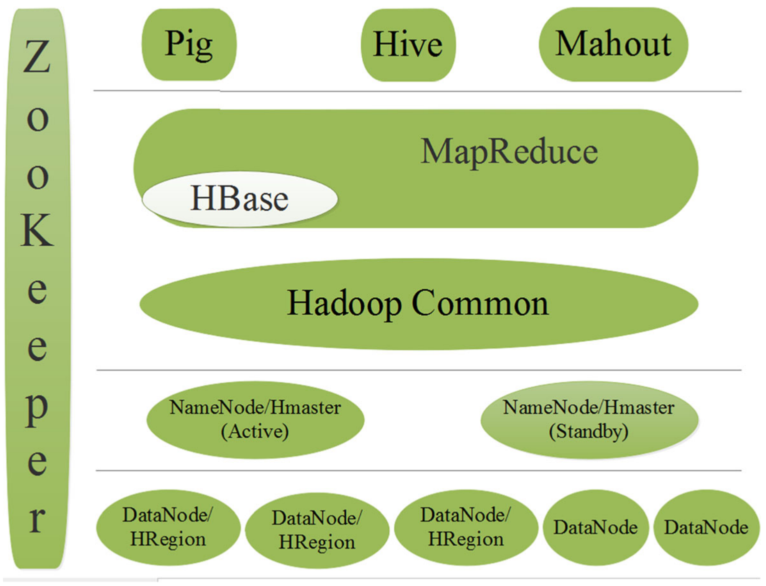

Fig. 2 Hadoop architecture

$$Va_{ij} = \left[\begin{array}{ccccc} & O_1 & O_2 & \cdots & O_n \\ U_1 & Va_{11} & Va_{12} & \cdots & Va_{1n} \\ U_2 & Va_{21} & Va_{22} & \cdots & Va_{2n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ U_m & Va_{m1} & Va_{m2} & \cdots & Va_{mn} \end{array}\right] \tag{1}$$

3.4 Primitive HBase database storage

As for the technology of Primitive HBase data storage specific optimization, data sharing and data distribution are the two ordinary optimization strategies. This paper chooses the latter, because although data sharing unifies the manage storage of all types of data and solves the congestion problem when a single queue is used to accept data, it neglects the different patterns among the multi-source data thus will impede the efficiency of inputting and data stream query. Whereas data distribution optimization is used, consistent hashing algorithm can conquer the monotony of the traditional Hash data distribution algorithms, and then solve the problem of the high load of data server [15, 16]. Meanwhile, proceeding from the actual demand of the deployment of data communication software, the adoption of writing parallel data based on the multi-threading technology can better solve the problem of delay processing of data communication.

HBase stores data in table form, which consists of rows and columns. Each column belongs to a specific series of columns—identified by rows and columns in the table as elements, the storage unit of each element is used to save multiple versions of the same data, identified by a timestamp. Table 1 illustrates the network www.cnn.com data storage logical view.

In addition to the logical storage model, Table 2 shows the physical storage model of HBase. HBase physical model is stored in the column of sparse matrix row/column, and it is actually a line integral of the conceptual model and stored in accordance with the column.

4 Wireless network data real-time storage model

The multi-source, heterogeneous, massive and high-speed data storage structure is designed for data access preprocessing area, the function of which is to integrate and normalize data operations in order to ensure the data is integrated, effective and able to meet the buffer to queue; Data buffers (which maintain different types of data objects in real-time processing to improve the speed of the writing data batch); Data scheduling write area (whose function is

Table 1 The logic view of data storage

Line keyword	Time stamp	Contents	Anchor	Mime
com.cnn.www	T9		"Anchor:cnnsi.com" "CNN"	
	T8		"Anchor:my.look.ca" "CNN.com"	
	T6	"<html>…"		"text/html"
	T5	"<html>…"		
	T3	"<html>…"		

Table 2 The physical storage form of data

Line keyword	Time stamp	Contents
com.cnn.www	T5	"<html>…"
	T4	"<html>…"
	T3	"<html>…"
com.cnn.www	T8	"Anchor:cnnsi.com""CNN"
	T7	"Anchor:my.look.ca""CNN.com"
com.cnn.www	T6	"text/html"

to receive fragmented data based on multi-threading technology, and use parallel write method to store data); Data storage area (whose function is to send massive data organization and distribute the information stored in the database based on the distributed storage write area of the HBase database. The HBase cluster logical isolated storage provides different rights for different users with different types of data). The structure is shown in Fig. 3.

4.1 Multiple data storage buffer

The structure of multi-source buffer is shown in Fig. 4. The pre-processing area will be accessible after the multi-source data conversion queue transferred to the buffer zone through the hash method. In order to improve the rate of data transmission after partitioning, the queue of data is stored in a linked list structure. There are three queues in each buffer area waiting for data splitting. The organization of the data partition is shown in Fig. 5.

Faced with nearly one million high-speed data per second, the system's write buffer speed is set to 6 MB of space. The completion of the queue after the completion of the window threshold call HBase multi-threaded write method to send data to the buffer, which allows the client to call request execution through remote procedure, and the data will be sent to the server for permanent storage.

4.2 Filters and coprocessors

Users can filter data on specific storage cells in multiple dimensions (rows, columns, data versions) in HBase,

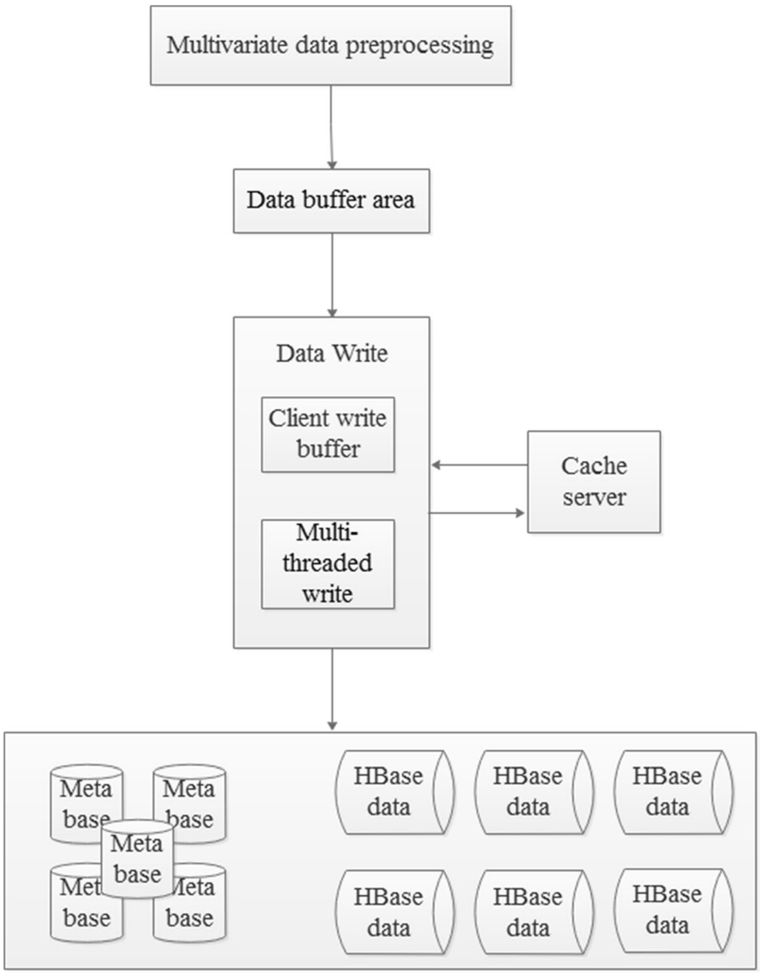

Fig. 3 Real-time system structure

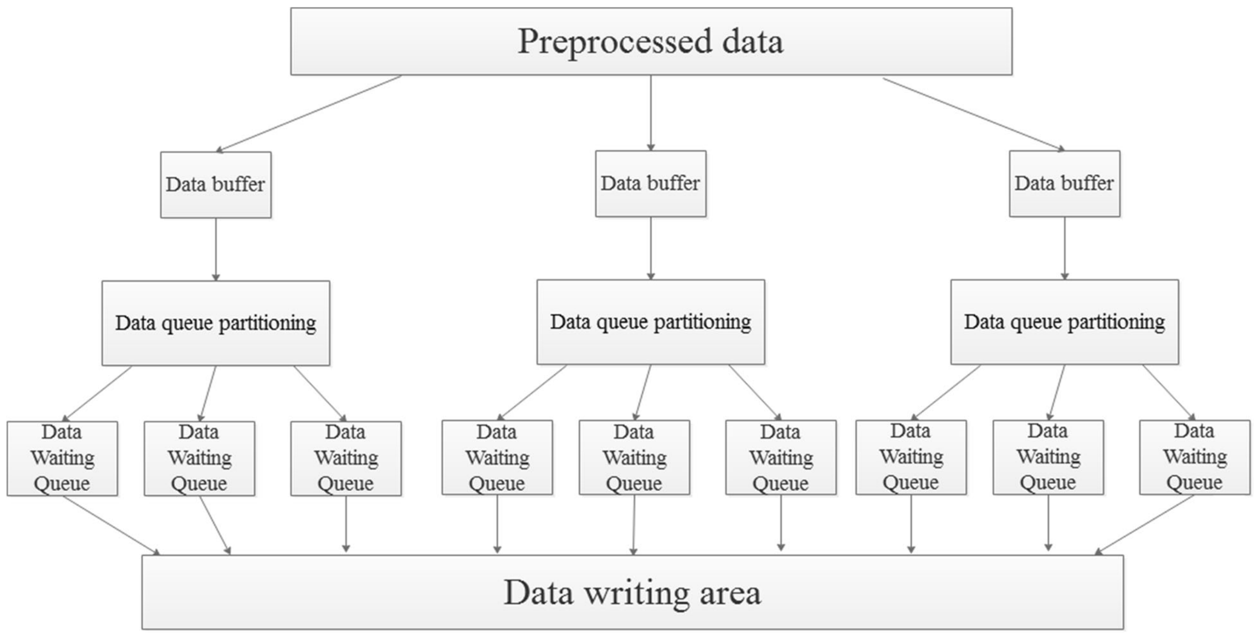

Fig. 4 Multi-source buffer structure

significantly improving the efficiency of retrieving data from the table. When transmitting large amounts of data, HBase provides the Endpoint coprocessor that deals with the distribution of complex code to the various servers (Region Server) to perform computational tasks, and then return the results to the client. Compared with the method of retrieving data from the data server, this method has the advantages of reducing the pressure of network transmission and client computing, and improving the efficiency of processing. This process is shown in Fig. 6.

5 HBase cluster configuration

First, in consideration of the performance conditions such as compression ratio, compression rate, and decompression rate, we expect to balance the I/O and CPU and select Snappy algorithm when cluster configuration is in compressed mode. Second, in order to reduce the buffer on the server, the client memory consumption and the number of RPC connections, it will be set by HTable client (Write buffer Size) to reduce the frequent communication with HBase and frequent I/O. Third, based on the consistency maintenance function, if the

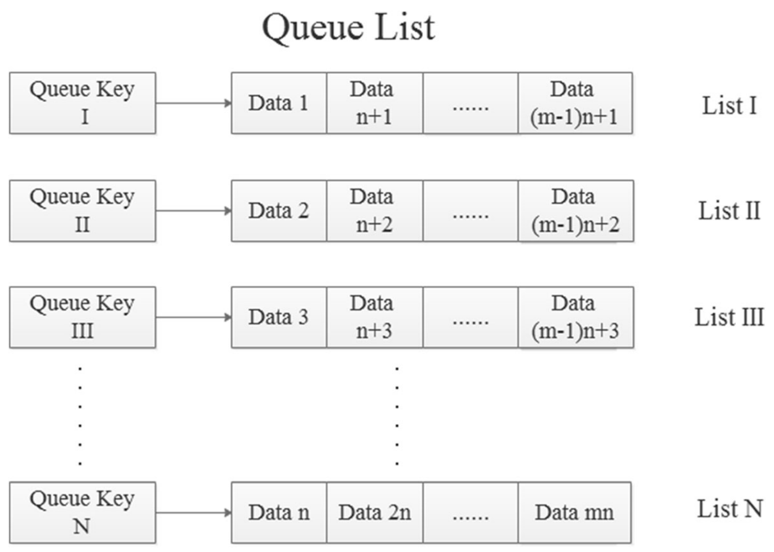

Fig. 5 Data organization structure

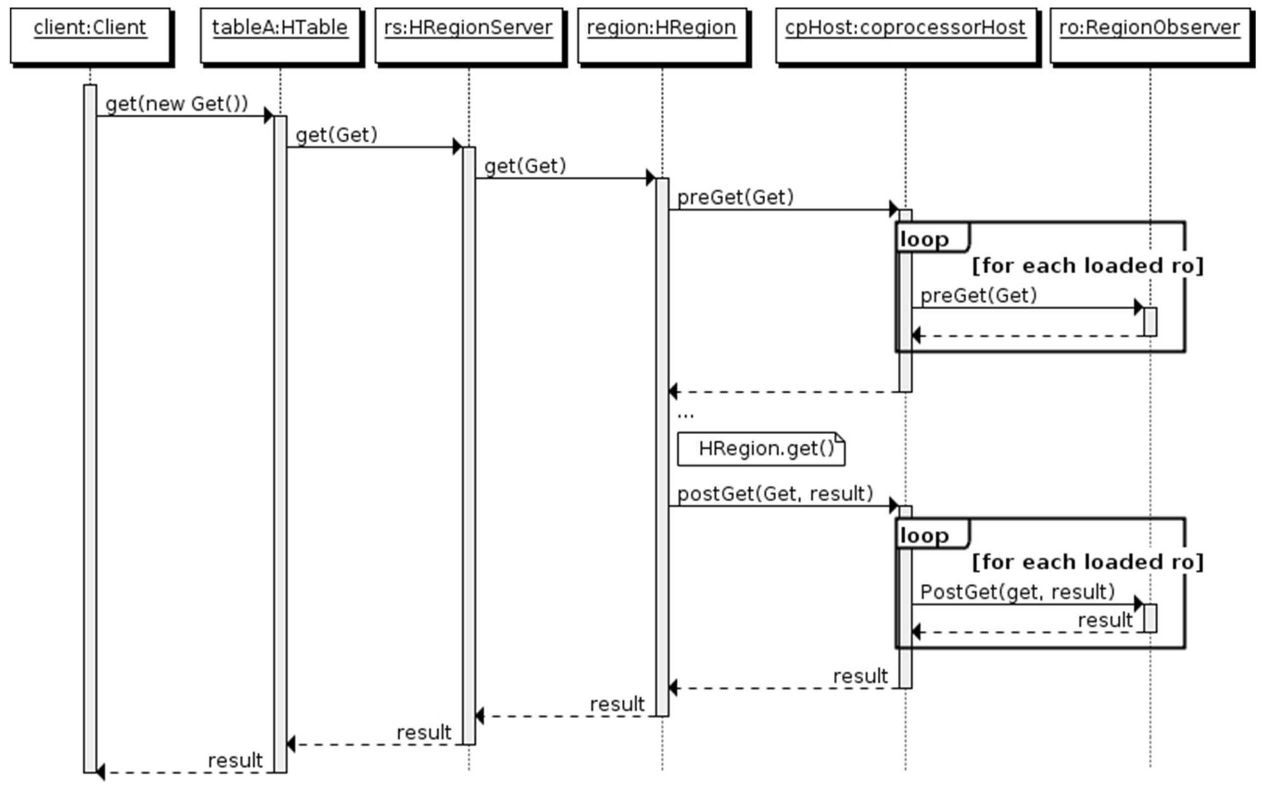

Fig. 6 The coprocessor transfer the process

amount of data returned from the server side is too massive or the data to be accessed is not in the cache, the calculation time increases, which will result in an error in the lease time. Therefore, the cluster configuration will adjust the parameter time (HBase.rpc.timeout). Finally, the server-side read each time the number of remote calls (scan.caching) settings will be adjusted. Too little adjustment will reduce the access efficiency, and too much will cause the memory to be occupied to reduce the access rate.

5.1 The system stores flow

When data access to the data pre-processing area, different types of source data are set up with a unique mark that discards incomplete data objects, non-standard data object marks, or formatted standardized operations, to ensure complete and efficient transfer of data to the buffer for queue partitioning. The multi-data buffer receives the preprocessed data to initialize the queue of the stored data, calculates the received data according to the timestamp attribute, and sends the data to the corresponding queue for temporary story. When the number of the buffer data reaches the threshold, the system will call the data write according to the pre-partitioning strategy, store the data objects in the corresponding database server, and update the count record table in the HBase cluster database at the same time. This process is shown in Fig. 7.

5.2 System storage algorithm implementation

The program execution initializes the data queue, according to the time attribute of the message to its hash calculation, and then compares the data size and write threshold on the basis of the hash value sent to the corresponding data queue. The HBase database will be started only if the data size is larger than the threshold or the receiving time

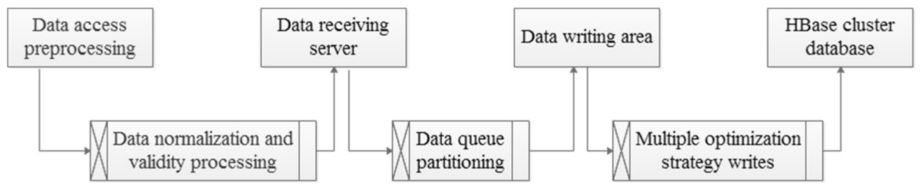
Data access preprocessing
Data receiving server
Data writing area
HBase cluster database
Data normalization and validity processing
Data queue partitioning
Multiple optimization strategy writes

Fig. 7 The system stores the flow

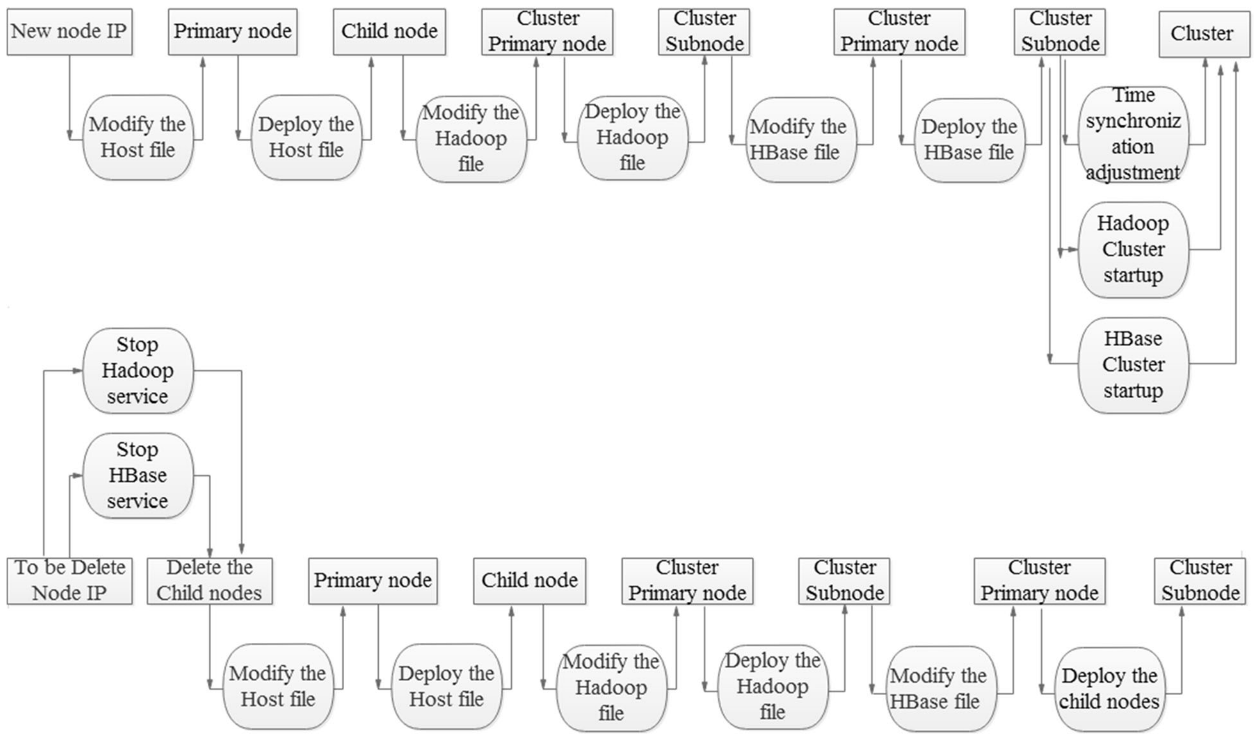

Fig. 8 HBase cluster node dynamic update process

is more than 1 s. The number of records based on the original write count is now the total number of records in the table.

5.3 Dynamic updating of HBase cluster nodes

When the amount of user data increases, the existing cluster database cannot meet the users' demands for space and performance, or when the system lost a lot of historical value of the application data, the server costs need to be reduced. If we manually modify the Hadoop and HBase file configuration, it is not only a waste of time and manpower, but also has high error rate in the process of modifying the configuration, so we use shell scripts in the form of cluster nodes for dynamic updating. This process is shown in Fig. 8.

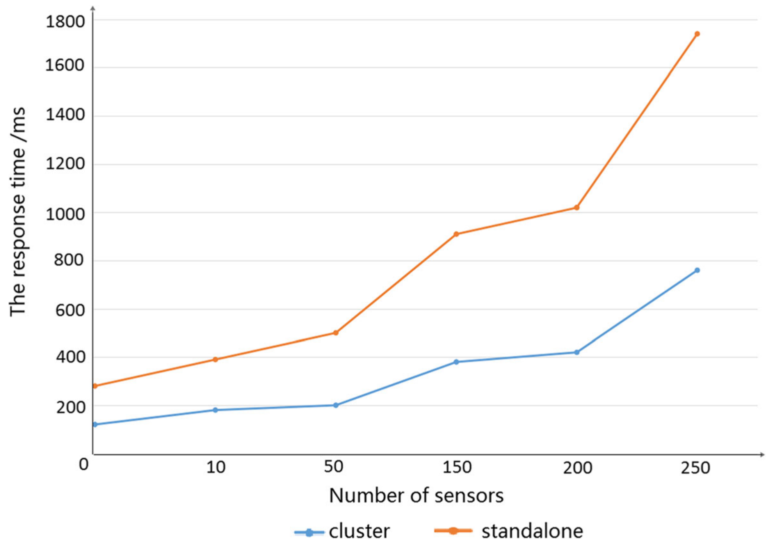

Fig. 9 The comparison of number of sensors and the writing time

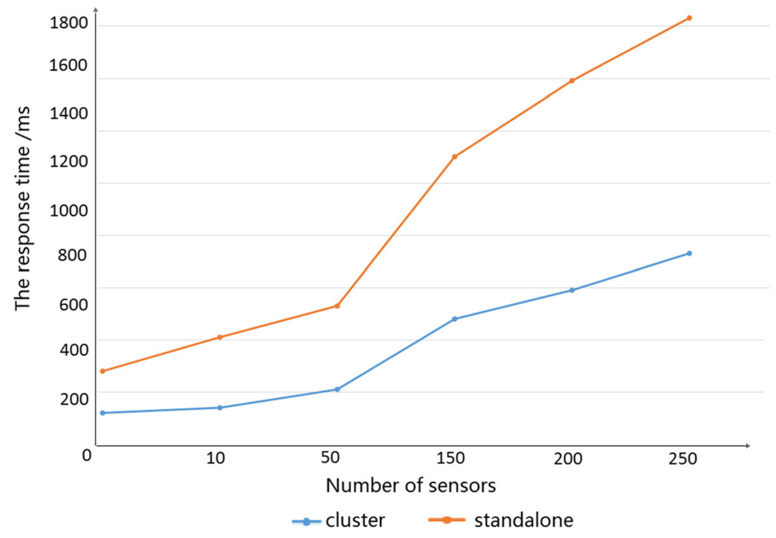

Fig. 10 The comparison of number of sensors and the read time

6 Performance analysis

Considering that most operations are writing and inquiry in the actual practice, the optimization of reaction time in this paper is based on the main performance analysis of the HBase data storage. HBase is a column-oriented database, and the applying column is the primary key. In addition, column alignment according to the dictionary sequence and the index is built according to B^+ tree on the data column. Therefore, the time complexity of data writing queries is $\log N$. When the query accesses, the client-side will first visit the major nodes, and then the write and read requests of the client will be responded through the guidance of major nodes and the coordination of area servers. Therefore, the time of inquired access is composed by the major nodes time T_{master}, the guidance of major nodes, and the coordination of regional server time $T_{\text{Area Server}}$. Set the number of inquiry data as i column, the number of regional servers as m coordinated by the major nodes, the data stored in each regional server as j column, the length of major nodes' content as i/j, the length of saved data in regional servers as $i/(j*m)$, and the total time is T_{Total}. Thus, the data access time is obviously shortened after optimizing through primitive HBase data storage.

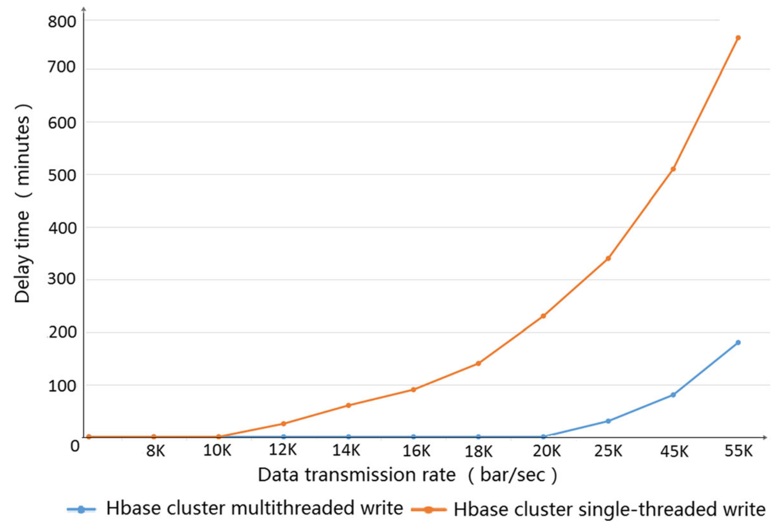

Fig. 11 Delay test simulation

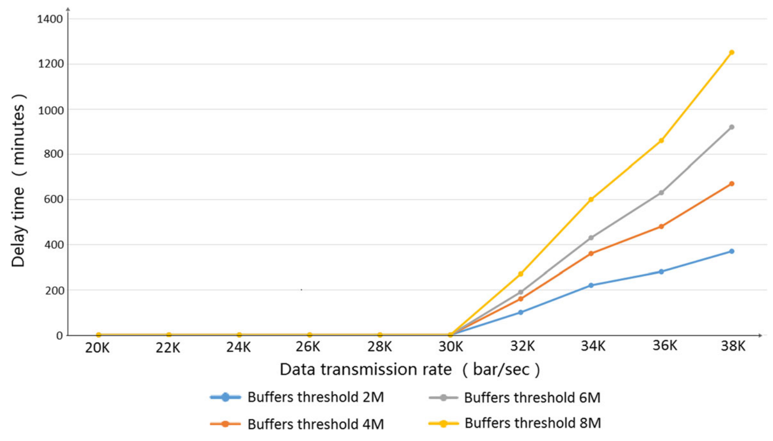

Fig. 12 Buffers threshold test simulation

$$\begin{aligned} T_{\text{Total}} &= T_{\text{master}} + T_{\text{AreaServer}} \\ &= \log\left(\frac{i}{j}\right) + \log\left(\frac{i}{j*m}\right) < \log(i) \end{aligned} \tag{2}$$

To verify the efficiency of storage, query, and extensibility of HB, we use 300 sensor nodes, which are single and configured of five node for the cluster to write and read time. Figure 9 shows a single cluster, and its relation with the sensor nodes is write time. As can be seen from the figure, the cluster write time is significantly shorter than the single write time, so it is of high efficiency. Figure 10 shows a single cluster, and its relation with the number of sensor nodes is access time. It can be seen from the figure that with the increase in the number of sensors, the cluster read time is significantly shorter than the single load time, so it has the property of efficient reading.

The design of the real-time storage method is based on Oracle data storage, and the data storage test is carried out separately. In the system designed in this paper, single-thread and multi-thread writing are used for the operation inside the data writing area. The specific settings are as follows: Write client buffer width and set the value to 6 M, the buffer size is 2, 4, 6, and 8 and the unit is MB, the data transmission rate is 1K, 2K, 3K … 30K and the unit is bar per second, the window time threshold is set to 2 s.

As shown in Fig. 11, the Oracle's streaming data as the distributed real-time storage method, when the data transmission rate is 12,000 per second, the stored data delay and the delay time increase constantly with the increase of data transmission rate. When the system uses single-threaded test, the delay point is 25,000 per second, and when using multi-threaded technology, the database throughput increases significantly, with no delay at the transmission rate of 55,000 per second.

As shown in Fig. 12, the horizontal axis is the amount of data sent per second and the vertical axis is the delay time of the HBase cluster database stores streaming data in real time.

Throughput efficiency is 25,000 times per second with a delay, and the delay will be longer as the data transmission rate increases. Simulated tests with different buffer thresholds are optimal when the threshold is set to 6 MB, and increasing or decreasing the buffer threshold setting will increase the memory delay time for real-time data.

7 Conclusions

In this paper, an efficient real-time storage model is designed under the guidance of storage optimization strategy. The real-time flow of data will be quickly stored in the cluster database which can satisfy the multiple requirements of users for data storage performance. In addition, the historical stream data stored in the text form will be migrated to the HBase cluster database in high efficiency and high stability. The paper also discusses that when the storage space is insufficient or the storage space of the text system of the cluster is too massive, the efficient and simple dynamic updating of the HBase database cluster will help optimize the database space usage.

Acknowledgements
This research was supported by NSFC (61602346) and Tianjin Science and Technology Project-International Joint Research and Development Project (Sino-Finnish Cooperation) (14RCGFGX00847).

Competing interests
The authors declare that they have no competing interests.

Author details
[1]College of Computer and Information Engineering, Tianjin Normal University, 300387 Tianjin, China. [2]College of Electronic and Communication Engineering, Tianjin Normal University, 300087 Tianjin, China. [3]Department of Computer Science, The University of Arizona, Tucson, Pima County, AZ 85705, USA. [4]Tianjin Key Laboratory of Wireless Mobile Communications and Power Transmission, Tianjin Normal University, 300087 Tianjin, China.

Received: 15 November 2016 Accepted: 21 February 2017
Published online: 04 March 2017

References

1. K Sun, D Wu, *MPC-based Delay-Aware Fountain codes for live video streaming*. IEEE International Conference on Communications, ICC, 2016
2. RO Schmidt, Multiple emitter location and signal parameter estimation. IEEE Trans. Antennas and Propagation. **34**(3), 276–280 (1986)
3. K Sun, H Zhang, D Wu, H Zhuang, MPC-based delay-aware fountain codes for real-time video communication. IEEE Internet Things J (IoT). **99**, 1-1 (2016)
4. V Jacobson, DK Smetters, JD Thornton, et al. Networking Named Content. Conext'09 Proceedings of International Conference on Emerging Networking Experiments & Technologies, **55**(1), 1-12 (2009)
5. Z Li, K Liu, Y Zhao, Y Ma, MaPIT: an enhanced pending interest table for NDN with mapping bloom filter. IEEE Commun. Lett. **18**(11), 1423–1426 (2014). doi:10.1109/LCOMM.2014.2359191
6. M Gruteser, D Grunwald, *Anonymous usage of location based services through spatial and temporal cloaking*. ACM/USENIX MobiSys, 2003
7. JF Gu, WP Zhu, MNS Swamy, Joint 2-D DOA estimation via sparse L-shaped array. IEEE Trans. Signal Processing. **31**(5), 1171–1182 (2015)
8. K Sun, D Wu, Video rate control strategies for cloud gaming. J. Vis. Commun. Image Representation. **30**, 234-241 (2015)
9. Z Li, L Song, H Shi, Approaching the capacity of K-user MIMO interference channel with interference counteraction scheme, Ad Hoc Networks. (2016) doi:10.1016/j.adhoc.2016.02.009.
10. K Sun, B Yan, H Gharavi, *Lowcomplexity content-aware image retargeting*. IEEE International Conference on Image Processing 2012 (ICIP'2012), 2012
11. Ghinita G, Kalnis P, Skiadopoulos S. PRIVE: anonymous location based queries in distributed mobile systems. Proceedings of International Conference on World Wide Web (WWW' 07), Banff. 1- 10, (2007).
12. GF Riley, TR Henderson, *The ns-3 network simulator*. Modeling & Tools for Network Simulation, 2009, pp. 15–34
13. Z Li, Y Chen, H Shi, K Liu, NDN-GSM-R: a novel high-speed railway communication system via named data networking. EURASIP J. Wireless Commun. Netw. **2016**(48), 1–5 (2016). doi:10.1186/s13638-016-0554-z
14. X Liu, Z Li, P Yang, Y Dong, Information-centric mobile ad hoc networks and content routing: a survey. Ad Hoc Networks, (2016), http://dx.doi.org/10.1016/j.adhoc.2016.04.005.
15. T Ming, Q Wu, Z Guoping, H Lili, *A new scheme of LBS privacy protection*. IEEE Radar Conference, 2009
16. K Sun, B Yan, *Efficient P-frame complexity estimation for frame layer rate control of H.264/AVC*. IEEE International Conference on Image Processing 2011 (ICIP'2011), 2011