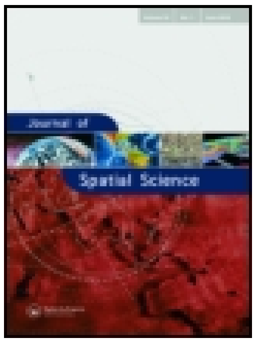


Journal of Spatial Science

ISSN: 1449-8596 (Print) 1836-5655 (Online) Journal homepage: http://www.tandfonline.com/loi/tjss20

Building an efficient storage model of spatial-temporal information based on HBase

Ke Wang, Guolin Liu, Min Zhai, Zhiwei Wang & Chuanyi Zhou

To cite this article: Ke Wang, Guolin Liu, Min Zhai, Zhiwei Wang & Chuanyi Zhou (2018): Building an efficient storage model of spatial-temporal information based on HBase, Journal of Spatial Science, DOI: 10.1080/14498596.2018.1440648

To link to this article: https://doi.org/10.1080/14498596.2018.1440648

Published online: 05 Mar 2018.

JOURNAL OF SPATIAL SCIENCE, 2018
https://doi.org/10.1080/14498596.2018.1440648

Building an efficient storage model of spatial-temporal information based on HBase

Ke Wang, Guolin Liu, Min Zhai, Zhiwei Wang and Chuanyi Zhou

College of Geomatics, Shandong University of Science and Technology, Qingdao, China

ABSTRACT
Geographic space is abstracted into a meta-semantic object (MSO), which is regarded as the smallest storage unit according to the semantic constraint principle and the characteristics of spatial-temporal information. An efficient organization and storage method is proposed by adopting the strong expansibility and real-time reading and writing characteristics of HBase through the encapsulation of MSO spatial-temporal information. This paper focuses on the classification and abstraction of geographic spatial-temporal information. The HBase platform is used as a carrier for the storage of massive spatial-temporal data. An HBase storage model is built by constructing a spatial-temporal data table and designing the physical structure of the MSO. By comparing the query time of the proposed model with that of the traditional spatial-temporal data model, the experimental results show that the query speed of the proposed model is higher. This provides a new idea for the organization and storage of massive spatial-temporal data.

KEYWORDS
HBase; semantic constraint principle; spatial-temporal information; meta-semantic object; query time

1. Introduction

With the rapid development of geographic information systems (GIS), spatial-temporal data are increasing dramatically (Ranjan *et al*. 2016). The construction of a smart city requires the effective organization and management of complex spatial-temporal information, such as coordinate information, text information and radio frequency identification (RFID). Consequently, more and more GIS applications have been used to build smart cities (Huang *et al*. 2016). Traditional spatial data storage systems are built by relational databases that are extended for spatial data. Generally, these systems are used to store structured data, but there is a certain lack of storage suitable for massive and unstructured data. Based on this, key aspects in the further development of GIS are the storing and processing of massive spatial-temporal data more clearly and completely (Wang *et al*. 2014, 2015, Deng *et al*. 2015, Ma *et al*. 2015). Therefore, the emerging Not Only SQL (NoSQL) provides technical support for the storage of massive and unstructured spatial-temporal data. The cloud platform (Karun and Chitharanjan 2013), which has high availability and real-time reading and writing characteristics that support massive data storage and management, plays an important role in

CONTACT Guolin Liu gliu@sdust.edu.cn

big data processing. As a cloud storage database, the Hadoop database (HBase) is a distributed, fault-tolerant, highly scalable, column-oriented, NoSQL database. It is an improvement of the Hadoop distributed file system (HDFS) (White and Cutting 2009) and creates a massively scalable and high-performance platform that handles heterogeneous data, including non-textual data types (George 2011). It provides a platform and method for users to manage GIS spatial-temporal data.

Recently, there have been studies on the organization of spatial-temporal data and HBase-based storage for it. Information for time, space and attribute has been organized by object-oriented technology in GIS (Gong 1997, Zhang *et al*. 2002). The state of geographic space and urban traffic data were organized and managed based on the changes in an object (i.e. the sequence of events) (Kathleen and Max 2000, Liu *et al*. 2003). A geographic spatial data model was constructed based on the characteristics and processes of spatial-temporal change (Albrecht 2005, Wu and Lue 2008). The data models mentioned above are specific for describing geographic spatial-temporal information, but a majority of these models are stored in relational databases. Relational databases have shortcomings that cannot be overcome in organization and storage of geographic spatial-temporal information. With the development of NoSQL and HBase, there is a new platform for the organization and storage of spatial-temporal information. For example, a type of massive image storage technology based on HBase solves the problem in HBase, which is a lack of parity and alignment of the byte arrays; this idea has been applied in urban monitoring systems (Zhu and Zhai 2013). A distributed storage scheme in HBase based on a Hilbert R-tree index for spatial vector data was proposed (Wang *et al*. 2013). Based on HBase, a two-layer storage framework was proposed to meet the needs of data storage in large-scale wireless sensor networks (Zhou and Chen 2012). Moreover, there have been a great number of HBase-based studies regarding spatial indexes. The storage and rapid query of image data are realized using the distributed storage index structure of massive remote sensing data in Hadoop (Lei 2010). A vectorized spatial data distributed storage model and a spatial index method are proposed based on HBase (Fan *et al*. 2012). The resource description framework (RDF) data storage model is based on HBase, includes eight types of query algorithms, and proves to be feasible (Papailiou *et al*. 2014).

Although there are an increasing number of applications for HBase, most of them are used in computer data processing. There are few applications in GIS regarding the large amount of spatial-temporal data and the efficient storage of spatial-temporal data. In fact, the existence of HBase provides storage schemes for spatial data. Excellent high-performance computing frameworks have also emerged for the derivation of hidden relations and factors from multidimensional big data (Chen *et al*. 2015). However, there is less research on the storage of geo-spatial information after the precise and detailed organization of the storage. In summary, with the development of the Internet (Web 2.0) (Cox 2008, Shin and Kim 2008, Iwata 2012), the volume of spatial-temporal data generation is of the order of hundreds of millions every day. Based on the shortcomings of traditional databases in terms of scalability and the real-time interpretation of massive multiple heterogeneous data, this paper analyses the basic frame and storage principles of HBase and abstracts geographic spatial-temporal information according to the characteristics of spatial-temporal data. Then, the packaging of geographic spatial-temporal information is accomplished based on HBase storage. An efficient organization and storage model for spatial-temporal information based on HBase is constructed.

Table 1. Logical view of HBase.

		Column family: c1		Column family: c2		Column family: cn	
RowKey	TimeStamp	Column	Value	Column	Value	Column	Value
r1	t7	c1:1	v1-1/1				
	t6	c1:2	v1-1/2				
	t5			c2:1	v1-2/1		
r2	t4					cn:1	v2-n/1
r3	t3	c1:1	v3-1/1				
	t2			c2:1	v3-2/1		
r4	t1			c2:1	v4-1/1		

2. Principles of HBase data storage

The storage of spatial-temporal data needs to meet the demands of high-speed data storage and queries, massive data storage, and high scalability and operability due to the inception of big data. HBase is an open-source application that stores structured, semi-structured and unstructured data by column on Google's Bigtable (Cheng *et al*. 2010). It is a foundation for the efficient organization of spatial and temporal data.

2.1. *Logical model of HBase*

With the characteristics of sparse data storage, persistence and multidimensional mapping, an HBase table represents a mapping relationship that can be used to locate specific data by row, row+time-stamp or row+column (column family: column modifier) (George 2011). HBase logically organizes the data into nested mappings, and its sparseness allows for white space when the data are stored. HBase has only one RowKey for determining the storage column. Row information includes RowKey, TimeStamp and family Information. A column family contains the contents of several columns. Therefore, a row can be expressed in this form (i.e. RowKey, TimeStamp, Column family). The logical view of HBase is shown in Table 1.

RowKey in the above table consists of any string with a length that does not exceed 64 KB, which is equivalent to the unique key of the primary table and cannot be empty. Each HBase RowKey corresponds to each input row data, and is divided into multiple regions according to RowKey. In addition, each Region Server manages one or multiple unused regions, then it implements the distributed read and write operations. All records in the table are scanned for access through a single RowKey or an activity-full table in RowKey Region.

The modes of 'column family: qualifier' are defined when the HBase table is created. Each HBase table consists of one or more column families, and each column family is composed of multiple columns. The number and type of columns do not need to be defined because the 'column family: qualifier' can be expanded dynamically.

TimeStamp represents the temporal characteristics of the data in HBase, and is expressed as 64-bit integers and is automatically assigned by the HBase system when writing data. The most recent data are at the top of the HBase table according to the written data order. The value of each data unit consists of RowKey, column family: qualifier and TimeStamp.

2.2. *Physical model of HBase*

The HBase data table for the logical model is composed of many rows, but HBase data are based on columns in the actual physical storage. The HBase data storage diagram from the logical model to the physical model is shown in Figure 1.

As shown in Figure 1, data in a logical model are stored from Row 1 to Row *n* logically (i.e. row-by-row), but are stored by column families in the actual physical storage. The value of each column family and its corresponding RowKey and versions are stored in an independent file, which stores the information for each row in different column families.

3. Geographic spatial-temporal data organization based on HBase

Given spatial-temporal data characteristics in large amounts, from multiple sources and with complex structures, which are combined with the storage structure of HBase, urban spatial-temporal entity information such as file data, map data and radio data are abstracted into multiple meta-semantic objects.

3.1. Object-oriented representation of geographic entities based on HBase

HBase can split the region automatically when data in the record table exceed a certain threshold; therefore, HBase tables are not limited in size. Spatial-temporal data using as much detail as possible and following a set of rules are described to meet the requirements of efficient organization and management.

Spatial-temporal entities are divided into MSOs, composite semantic objects (CSOs) and aggregate semantic objects (ASOs) according to the combination of semantic and spatial constraints. These entities consist of basic attributes, general attributes, geometric objects, method sets, non-spatial semantic relations and spatial semantic relations. They can realize the complete define of spatial-temporal data, describe the information of geographic entities in detail, and construct a spatial-temporal data model with object-oriented technology, as shown in Figure 2.

Definition 3.1: a geographic entity is a complete minimum logical unit that has an objective geometrical structure and shows dependent semantic attributes. Geographic entities have the following characteristics: (1) an arbitrary geographic entity is identified by a unique identifier (ID); (2) a set of attributes (e.g. type attributes and spatial attributes) is used to identify a unique geographic entity; (3) an MSO is characterized by temporal features, spatial features and attribute features, and there are different basic topological and metric relations among different MSOs; (4) according to the combination of the meta-semantic objects (i.e. the combined or aggregated relationship between similar MSOs), the entity-object-oriented

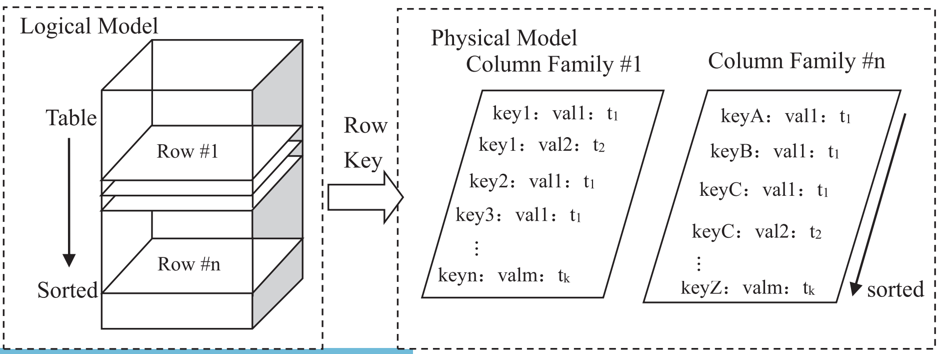


Figure 1. The HBase data storage diagram from the logical model to the physical model.

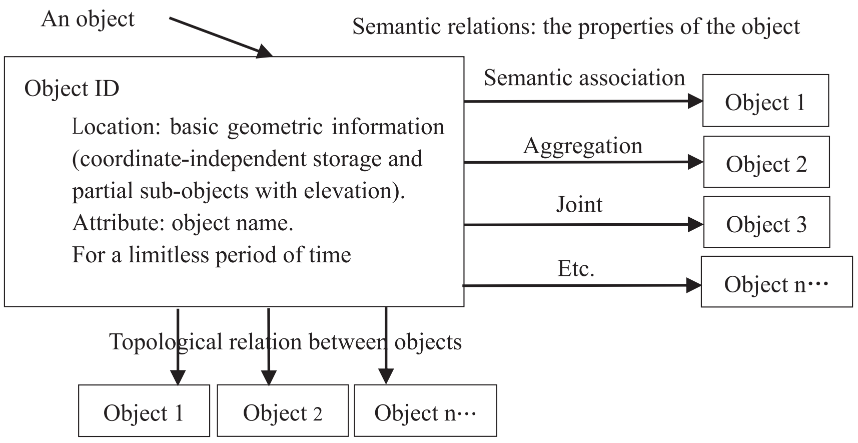

An object
Semantic relations: the properties of the object
Object ID
Location: basic geometric information (coordinate-independent storage and partial sub-objects with elevation).
Attribute: object name.
For a limitless period of time
Semantic association
Aggregation
Joint
Etc.
Object 1
Object 2
Object 3
Object n···
Topological relation between objects
Object 1
Object 2
Object n···

Figure 2. Object-oriented GIS data model.

technology of the geo-spatial data is abstracted as a combination of the semantic object and the aggregative semantic object. A geospatial entity is abstracted into a CSO and ASO combination of the object-oriented technology. The mathematical expression is as follows: $S = f(s_1, s_2, \cdots, s_n)$, where S represents geographic space, s represents a separate geographic entity, and geographic space S consists of several small geographic entities and their spatial relations.

(1) Description of the MSO

Definition 3.2: the MSO is a semantic object that has independent features in geographic space. An MSO has the following characteristics: (1) it can exist independently and (2) it cannot exist independently on the Earth's surface, but it needs to be attached to other objective geographic entities. An MSO is a basic unit for geographic entities, and a geographic entity is composed of different MSOs. An MSO serves as the RowKey in the HBase list table. This is expressed mathematically as follows:

$$MSO = \{Code, CN, OID, ON, \{A_i\}, G, \{OP_i\}, \{AR_i\}, \{SR_i\}\} \tag{1}$$

where *Code* represents the classification code of the MSO; *CN* represents the MSO classification information, such as architecture; *OID* represents the ID of the MSO; *ON* represents the standard name of the MSO (e.g. the Yangtze River); $\{A_i\}$ represents the common attribute set; G represents geometric features; $\{OP_i\}$ represents the method set; $\{AR_i\}$ represents the non-spatial semantic relationship set, which is composed of non-spatial expressions of semantic objects (e.g. how an educational building is a part of a school); $\{SR_i\}$ represents the spatial semantic relationship set, which is composed of the spatial relationship between MSOs.

An MSO is expressed through an object-oriented technology composite based on formula (1). An MSO describes spatial semantic relations, non-spatial semantic relations, geometric expressions and operations using basic semantic coding types. The structure of an MSO is shown in Figure 3.

(2) Description of the CSO

If geographic space is expressed only using separate MSOs, it will ignore the combination relationship among different MSOs and acquire an increase in trivial information. Therefore, several indivisible meta-semantic objects are expressed as a whole when combined with object-oriented technology. For example, a table is expressed in the form of combined semantic objects. These MSOs constitute a composite semantic object with the characteristics of spatial and non-spatial relationships.

Definition 3.4: a geographic entity composed of multiple MSOs has a part-whole relationship, but the whole and the part are a type of organic unity and are not the simple part sum. Therefore, the entity is defined as a CSO.

The parts of a CSO cannot be separated from the whole. A CSO not only has all the attributes and features of MSOs, but also has its own structure and space-time characteristics. Its mathematical expression is as follows:

$$CSO = \left\{Code, CN, OID, ON, \left\{A_i\right\}, G, \left\{OP_i\right\}, \left\{AR_i\right\}, \left\{SR_i\right\}, \left[MSO_i\right]\right\} \tag{2}$$

(3) Description of the ASO

Definition 3.5: the ASO is a super object composed of many types of geographic entities, such as MSOs, CSOs and its own attributes. The ASO, which takes a solid object in a complete geographic space as a whole, is not limited by layers or relative distance and realizes the overall expression of real geographic entities. Its mathematical expression is as follows:

$$ASO = \left\{Code, CN, OID, ON, SL_i, \left\{A_i\right\}, G, \left\{OP_i\right\}, \left\{AR_i\right\}, \left\{SR_i\right\}, \left[MSO_i\right], \left[CSO_i\right]\right\} \tag{3}$$

In the above formula, SL_i represents the spatial level of the feature space where the ASO is located.

3.2. Expression of the spatial relationship of semantic objects

Geo-spatial cognition research mainly includes the process of geo-spatial perception and the process of geo-spatial thinking information. The geo-spatial information cognitive model converts information from the real world into simulated space and describes geographic space according to the geographic position of geographic things (i.e. where) and the nature

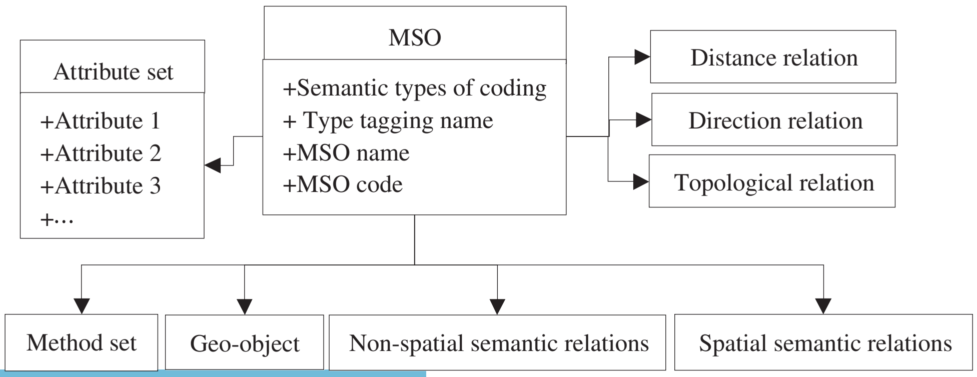


Figure 3. MSO structure and attributes.

of geographic things (i.e. what). The unique RowKey is determined by a semantic object, which can save all types of information regarding geographic entities and preserve semantic relationships between entities (Wu *et al*. 2015).

Geographic space is described by geo-spatial entities in the whole space. Its mathematical description is as follows:

$$S = \{E, R\} \tag{4}$$

In the above formula, $E = \{e_1, e_2, \cdots, e_n\}$ represents a collection of geo-spatial entities $e_i (1 \le i \le n)$, and R represents the relationship between geo-spatial entities (e_i).

Each MSO uses the unique information identifier (ID) to distinguish itself from other MSOs, and it is also used to determine the RowKey. The ID of an MSO has uniqueness and invariability. The spatial and attribute information of the MSO changes over time. The expression of the spatial relationships between MSOs is as follows:

$$MSO = \{u_{ID}, S(t), P(t), T(T_s, T_d), A\} \tag{5}$$

In formula (5), u_{ID} represents a unique identification code of the MSO in geo-spatial space; $S(t)$ represents the spatial features of MSO changes over time; $P(t)$ represents a set of attribute characteristics of the MSO; $T(T_s, T_d)$ represents the start and the end times of the MSO changes; T_s, T_d represents the valid time and transaction time, respectively; A represents the time, space and the attribute operations of the MSO.

$$S(t) = \{(\overrightarrow{p_1}, t_1), (\overrightarrow{p_2}, t_2), \cdots, (\overrightarrow{p_n}, t_n)\} \tag{6}$$

$$P(t) = \{(A_1^1, A_2^1, \cdots, A_m^1, t_1), (A_1^2, A_2^2, \cdots, A_m^2, t_2), \cdots, (A_1^n, A_2^n, \cdots, A_m^n, t_n)\} \tag{7}$$

In formula (6), $\overrightarrow{p_1} = \{p^1, p^2, \cdots, p^n\}$ represents a set of spatial data for an MSO. In formula (7), A_j^i is the jth attribute characteristic of an MSO at moments t_i, $j \in [1, m]$, and $i \in [1, n]$.

3.3. Organization and storage processes of geographic spatial-temporal data

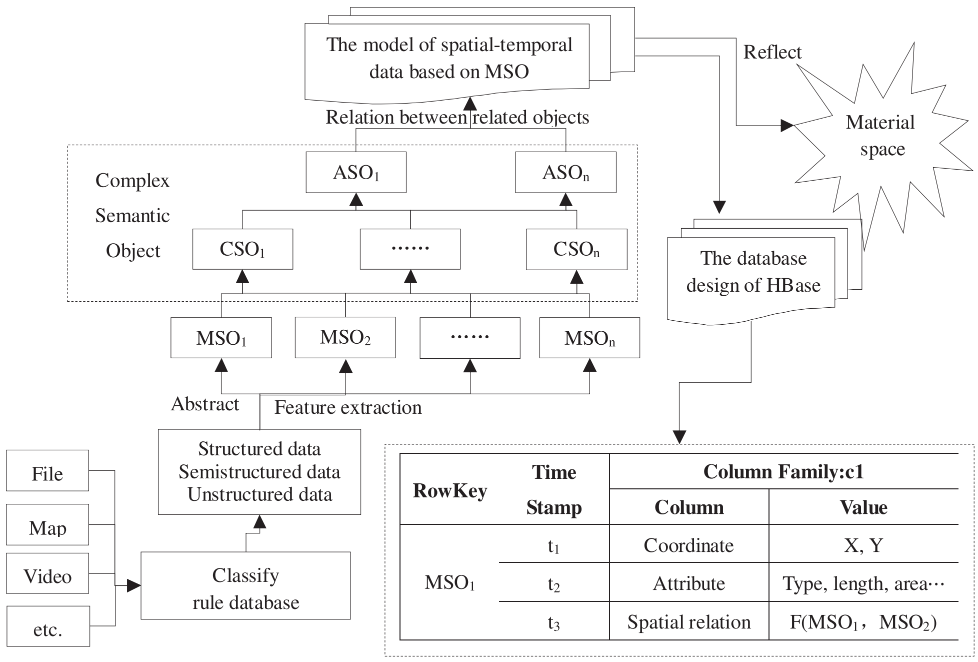
Spatial-temporal data organization is actually an abstract expression in the HBase environment (Long *et al*. 2011). When modelling geographic space, an MSO is the smallest object regarding organization and storage. It further packages the attributes and operations of CSOs and ASOs to avoid data redundancy and improve data transfer and development. Spatial-temporal semantic object organization based on HBase is shown in Figure 4.

Based on Figure 4, the object is represented by the bridge between the real world and the organization model in complex geographic space.

4. An efficient spatial-temporal information model constructed based on HBase

4.1. The efficient storage process of spatial-temporal data based on HBase

HBase is organized using a master/slave framework inside a distributed framework based on the efficient organization of geo-spatial information (Ding 2014). Spatial-temporal



RowKey	Time Stamp	Column Family:c1	
		Column	Value
MSO₁	t₁	Coordinate	X, Y
	t₂	Attribute	Type, length, area···
	t₃	Spatial relation	F(MSO₁，MSO₂)

Figure 4. Organization process for spatial-temporal data based on HBase.

information is classified and sorted by the HBase system and stored by a physical medium in the form of an HBase table. The spatial-temporal object that corresponds to the data of an MSO is stored on the same server or on a different server. The scalability of HBase is defined by a region that can split automatically when the region data are accumulated to a certain extent. After splitting, two sub-regions are assigned to different region servers by multi-threaded parallel writing to the HBase storage system. The efficiency of the HBase storage and its management of geo-spatial data is improved by MapReduce computations (Eltabakh *et al*. 2010, Feng *et al*. 2011, Kang 2011). The storage process based on the geographic data of HBase is shown in Figure 5.

4.2. *HBase storage module design*

In the data visualization module, HBase can solve the problem of massive and heterogeneous data storage, and it can use MapReduce calculations to improve the HBase data storage system (Ren 2011, Liu *et al*. 2013). There are often two types of spatial data that appear when performing calculations on data. One is the temporary data of spatial-temporal information of mobile geographic entities, which changes frequently. The other is the amount of data that is increasing, which will be applied less in the future. The use of MapReduce calculations can effectively solve the above two types of problems. During the the design of the data storage module, the MapReduce calculation program is often used to address the file in the HDFS and convert it to an HBase file type, which then uses an open-source program to input the HBase data. The frame diagram of the data storage is shown in Figure 6.

HBase is stored in an HDFS distributed storage system. HDFS is responsible for data storage and platform compatibility issues. In addition, the combination of Hive, Pig and other

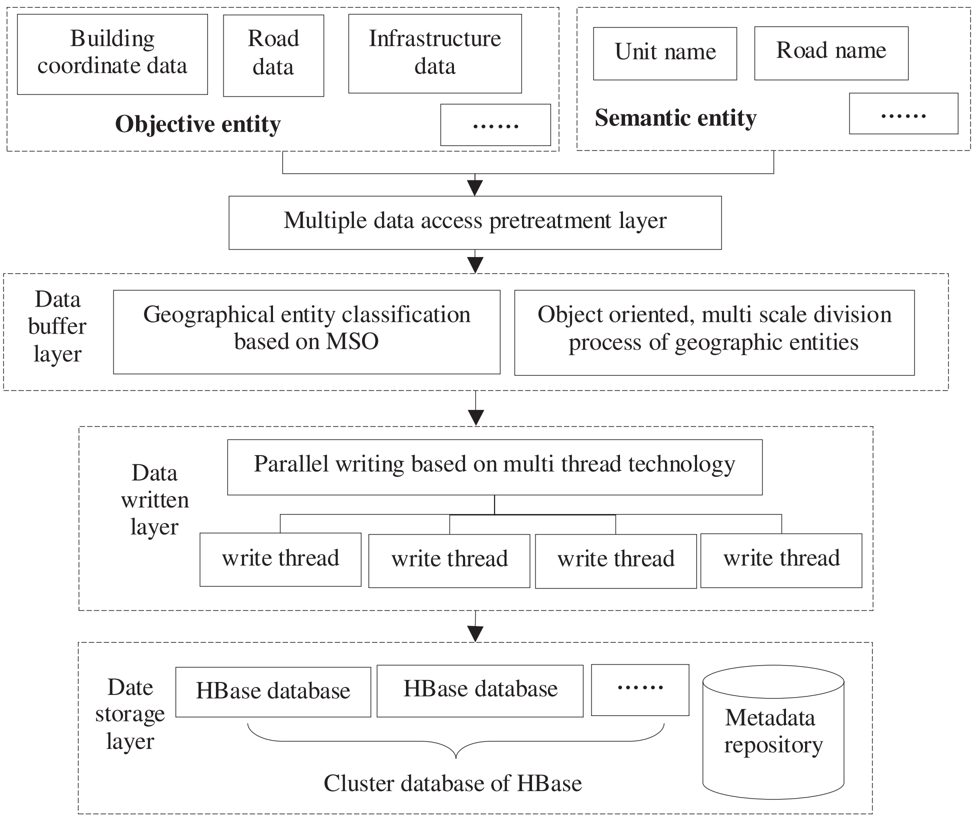


Figure 5. Distributed storage process of geographic information.

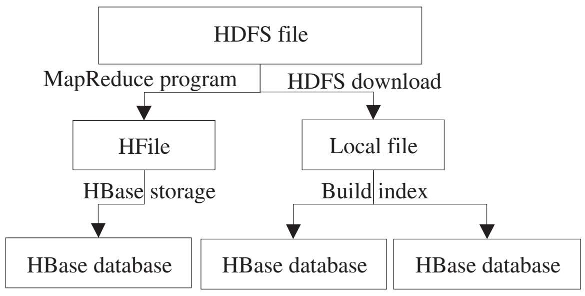


Figure 6. Frame diagram of the data storage.

HBase tools in the Hadoop platform satisfies the requirements of data storage and the demands of analysis. Specifically, regarding the increasing and massive quantities of structured data, unstructured data and semi-structured data in geo-spatial space, HBase makes good use of the characteristics in the MapReduce computation, the splitting of regions and the unconstrained and extensible column storage. It realizes the storage and management of these massive, multisource heterogeneous data.

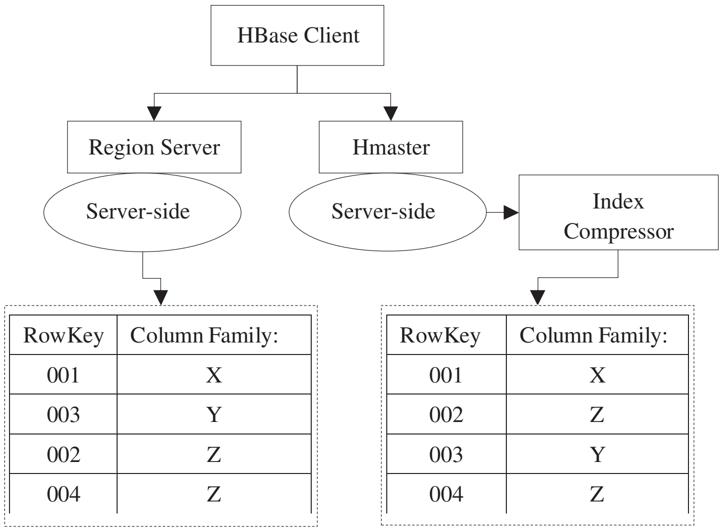


Figure 7. Two-level index overall architecture diagram.

Table 2. The overall structure of the geo-spatial information of the city.

		Column family		Column family	
RowKey	Stamp time	Attribute	Value	Composition	Value
City name ID		Attribute:Name	the name	Composition:Roads	road1, road2···
VT		Attribute:Type	the type	Composition:Facilities	station1···
		Attribute:Address	the address	Composition:Buildings	building1···
		Attribute:Area···	the area···	Composition:Moving Objects	vehicle1···

5. Experimental analysis

5.1. Application of proposed data model

The model proposed in this paper aims to store the geographic information in space with detail and precision. This paper takes the road information of a city as an example to describe the process of information storage and compare it with the traditional data storage method. The HBase physical storage table is designed for every spatial-temporal MSO abstracted from the geographic entities in a city. The HBase table is divided by units according to the HBase column storage characteristics, and different types of object tables are established in the geo-spatial space of a city. The overall structure of geo-spatial information of a city is shown in Table 2.

In Table 2, the ID of the city name is used as the RowKey in the HBase storage, and the TimeStamp is the version time in the established table. The HBase storage table of the urban geographic space mainly includes two families: composition and attribute. The composition family represents the composition of the urban geographic entities, including a collection of geographic entities, such as city roads, buildings, facilities and other entities. The attribute column family stores attribute information of the MSOs, CSOs or ASOs, including the name, area, type and other attributes of the buildings. For example, the HBase storage table of a road is shown in Table 3.

From Table 3, the road ID is identified by the RowKey in order to ensure each row of data is unique. The attribute and segment column families were designed to express the road clearly. Attribute describes the road attributes, such as name, type, length and width. Segment describes the road segment information and the value of the road information.

The road is composed of numbers for each section. Each section can be used as an MSO for organization. The HBase storage table of a road segment is shown in Table 4.

Table 4 shows the RowKey that represents the unique identifier for each row in the linked HBase storage table. The attribute column family is similar to the road attribute information (e.g. name, type, length and width of the road section). The MSO column family identifies the spatial information of the respective road section. The value describes the unique information of an MSO. In addition to the road, other entities in urban geo-spatial space are also represented according to formula (1).

5.2. Two-level index query of spatial-temporal data based on HBase

The query for massive spatial-temporal data based on the RowKey uses two-level index query technologies in HBase from the spatial-temporal storage model. The key words and corresponding spatial-temporal information are input in a client window, and the index of the input content is realized using a parallel computing framework with MapReduce programming. On the same region server, only one connection with the region server is established, which not only reduces the complexity of the connection, but also improves the query speed.

The HBase architecture of the two-level index query is shown in Figure 7.

The column family: qualifier is used as the Index because HBase is stored according to the column family. The index value is stored as original RowKey in the HFile and written in a specific folder corresponding to a Region using StoreFile.

First, a user sends the query request, and the master server reads the query service requirements and builds the index. Second, regarding the server side, a two-level index for management is established in the indexing compressor simultaneously. Each region on the region server side provides a call-back function for the client data to obtain, write, delete, scan and perform other operations in the indexing compressor. Finally, the call-back function is overloaded to achieve efficient organization, storage and query of the geo-spatial data. The example and the index values corresponding to the example are shown in Table 5.

As shown in this table, the index table is related to the RowKey and the column family: qualifier. The table is arranged on the order of RowKey-Value in the original table, and the order is Value-RowKey in the index table. As shown in Table 5, there are the specific example and the index values corresponding to the example. The following key value pairs are stored

Table 3. The HBase storage table of a road.

		Column family		Column family	
RowKey	Stamp time	Attribute	Value	Segment	Value
Road ID	VT	Attribute:Name Attribute:Width Attribute:Length···	the name the width the length···	Segment	Segment1, Segment2, Segment3···

Table 4. The HBase storage table of a road segment.

		Column family		Column family	
RowKey	Stamp time	Attribute	Value	MSO	Value
Road Section ID	VT	Attribute:Name Attribute:Type Attribute:Width Attribute:Length···	the name the type the width the length···	MSO	LID1 LID2 LID3 ···

Table 5. The example and the index values corresponding to the example.

Example table		The index value corresponds to example table	
RowKey	Column family:qualifier	RowKey	Column family:qualifier
Value 1	Row 21	Row 21	Value 1
Value 1	Row 23	Row 22	Value 2
Value 2	Row 22	Row 23	Value 1
Value 2	Row 25	Row 24	Value 3
Value 3	Row 24	Row 25	Value 2

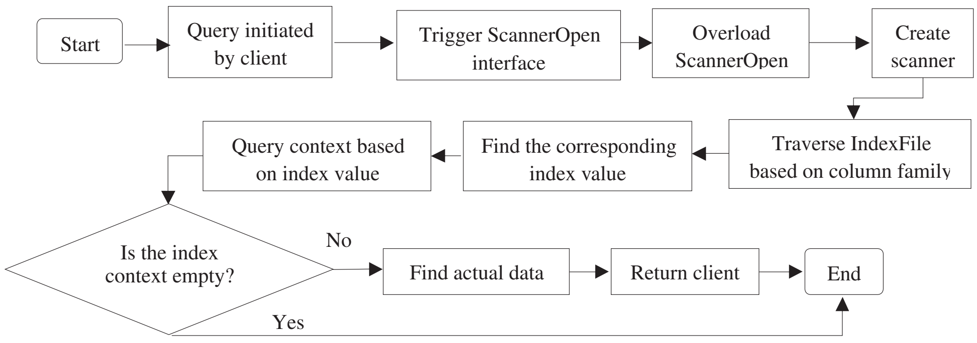

Start
Query initiated by client
Trigger ScannerOpen interface
Overload ScannerOpen
Create scanner
Traverse IndexFile based on column family
Find the corresponding index value
Query context based on index value
Is the index context empty?
No
Find actual data
Return client
End
Yes

Figure 8. Index query flow chart.

in the HFile: (Value1, Row21), (Value1, Row23), (Value2, Row22), (Value2, Row25), (Value3, Row24).

A concrete application when querying the index from the main table is shown in Figure 8.

The process is as follows: the region calls the open scanner when the client initiates a query interface; the region observer can capture this event and issue a notice to the interface, then the index table query is implemented by the function. If the query context is empty, it returns null and needs to use a full table scan query; if it is not empty, the real data table record is queried and the result is returned to the client using the region scanner. The implementation process of the two-level index query of GIS spatial-temporal data is shown in Figure 9.

As shown in the above figure, the query contents are matched with the keyword based on the HBase spatial-temporal data storage model, where GIS spatial and temporal data have efficient organization, and the query structure is imported into the user's requested HBase data table after the MapReduce calculations have been performed; then, the data search operation is realized.

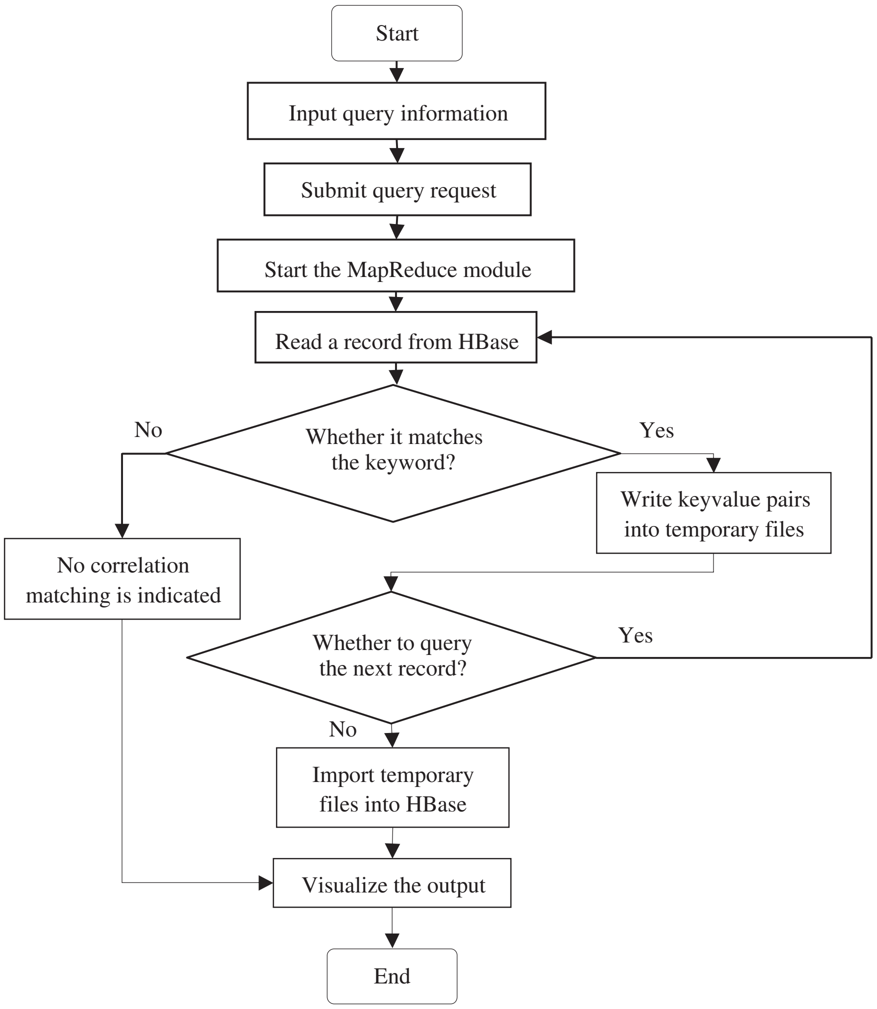

Start
Input query information
Submit query request
Start the MapReduce module
Read a record from HBase
Whether it matches the keyword?
No
Yes
Write keyvalue pairs into temporary files
No correlation matching is indicated
Whether to query the next record?
Yes
No
Import temporary files into HBase
Visualize the output
End

Figure 9. GIS spatial-temporal data query implementation process.

5.3. Experimental environment and result

The experimental environment is constructed in the Hadoop 2.4.0 distributed cluster. Five servers under the same gigabit switch are used for the experiment. The configuration of each server includes an Intel four-core 2.3 GHz processor with a 4 GB memory, 500 GB hard disk space and a CentOS 7.0 operating system. The HBase version is HBase 0.98. Meanwhile Oracle 10 g and ArcSDE spatial databases are established using the same configured server.

This study used urban road data in Guilin, China, as an example. The size of the data is approximately 1.9 GB and they contain 957,460 surface vectors. Each data vector contains 20 fields, such as the location of the road, the time of construction and the administrative area. The data are stored in the Oracle database and the HBase database. First, 200 rectangular areas with different sizes are set up, where the size of each rectangle contains different amounts of data. Then, the spatial query experiment in Oracle is performed and the database is distributed, subsequently, by utilizing the client for the simulation of multiple users. The concurrency level varies when using more threads. The model runs every 20 min, continuously, under different concurrency levels and calculates the average time for each spatial query. The temporal consumption results in two system architectures at different concurrent levels, as shown in Figure 10.

The results show that the spatial query efficiency in the distributed environment is much higher than that in the single environment. Specifically, when the number of concurrent levels increases, the impact of the concurrent query efficiency increases, but the impact on a distributed environment is lower than that on a single environment. The main reasons are as follows.

(1) HBase is a memory-based data-access mechanism. The data are stored in the memory after reading the data table file into HDFS, and the memory access speed is much higher than with the disk; therefore, the efficiency of the memory access has greatly improved compared to the Oracle method, which uses disk access.
(2) A distributed environment reduces the system load. Spatial querying is a data-intensive operation, and the largest part of the system load is the I/O operation of the disk file. In a distributed environment, the disk I/O pressure is distributed to each database server, which causes the impact of the concurrent operation on the overall efficiency to be lower than that with the single environment. The number of concurrent requests in the overall system is greater than that in the single environment.
(3) Spatial elements are stored in rows, and each row stores the value of a field in the HBase. The number of records that need to be retrieved is reduced when no fields are queried, which causes the query to be faster than when querying all fields.

Time consumption during a concurrent spatial query in the two system architectures is shown in Figure 10, where (a) represents different spatial query times in the two system architectures; to give a clearer illustration of the gap between the two storage environments, (b) represents the difference value (D-value) of the spatial query times in the two system architectures. The details are as follows.

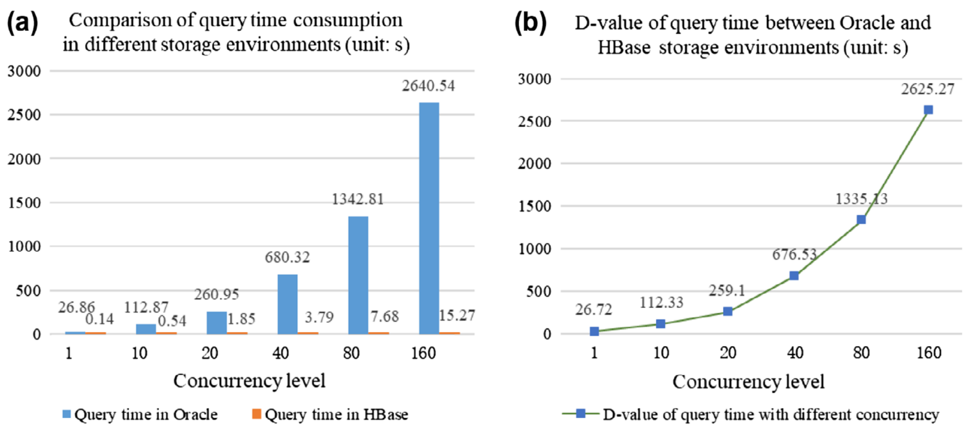


Figure 10. (a) Spatial query temporal comparison in two system architectures; (b) D-value of the spatial-temporal queries in two system architectures.

As shown in Figure 10(a), with an increase in concurrency, the query time of the two storage structures is increased; however, the HBase environment increases slowly, and the query speed is much faster than that of the Oracle environment. According to Figure 10(b), it can be seen that when the number of concurrency levels is higher, the gap between the two storage environments becomes more obvious. The experimental results show that a distributed environment can improve the storage and query efficiency of massive spatial data, and HBase guarantees a fast response when substantial concurrent data access is provided. It provides a solution to improve the data access efficiency of massive geographic spatial-temporal information.

6. Conclusions

Data storage based on HBase has been widely used in the computing field. An efficient organization method to manage spatial-temporal data is established by combining the principles of HBase with the characteristics of geographic spatial-temporal information. The key part is that an MSO is stored as the smallest unit in the HBase table. The physical structure of the HBase storage module is designed to realize the storage process of the spatial-temporal information. An efficient organization method for geographic spatial-temporal information based on HBase can not only represent the complexity of the GIS spatial-temporal information, but it also takes advantages of the HBase databases. It provides a new technical method that solves the management of spatial-temporal data when constructing a 'smart city'. However, with the development of the Internet, the complexity of the real world, and the diversity of users' needs, the complex information sharing and management methods and data security protection require further study.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work was supported by the National Natural Science Foundation of China [grant number 41404003], The Taishan Scholars Program of Shandong Province [grant number TSXZ201509], and the Graduates Innovation Fund of Shandong University of Science and Technology [grant number DKDYC180206].

References

Albrecht, J., 2005. Implementing a new data model for simulating processes. *International Journal of Geographical Information Science*, 19 (10), 1073–1090.

Chen, D., *et al.*, 2015. Fast and scalable multi-way analysis of massive neural data. *IEEE Transactions on Computers*, 64 (3), 707–719.

Cheng, Y., *et al.*, 2010. Research on large-scale data processing based on hadoop and relational database. *Telecommunications Science*, 11, 47–50.

Cox, A.M., 2008. Flickr: a case study of Web2.0. *Aslib Proceedings*, 60 (5), 493–516.

Deng, Z., *et al.*, 2015. Parallel processing of dynamic continuous queries over streaming data flows. *IEEE Transactions on Parallel & Distributed Systems*, 26 (3), 834–846.

Ding, C., 2014. Research on distributed storage and parallel query algorithm of spatial data in HBase. Thesis (Master). Nanjing Normal University.

Eltabakh, M.Y., *et al.*, 2010. Cohadoop: flexible data placement and its exploitation in hadoop. *Proceedings of the Vldb Endowment*, 4 (9), 575–585.

Fan, J.Y., Long, M., and Xiong, W., 2012. Research of vector spatial data distributed storage based on Hbase. *Geography and Geo-Information Science*, 28 (5), 39–42.

Feng, M., *et al.*, 2011. Terrain data diistributed computing using MapReduce. *Journal of Huazhong University of Science and Technology*, 39 (1), 24–27.

George, L., 2011. HBase : the definitive guide. *Andre*, 12 (1), 1–4.

Gong, J.Y., 1997. An object oriented spatio temporal data model in gis. *Acta Geodaetica Et Cartographic Sinica*, 4, 10–19.

Huang, F., *et al.*, 2016. PMODTRAN: a parallel implementation based on MODTRAN for massive remote sensing data processing. *International Journal of Digital Earth*, 9, 819–834.

Iwata, S., 2012. Big data era. *Journal of Information Processing & Management*, 55 (8), 543–551.

Kang, J.F., 2011. *Technologies of storage and efficient management on cloud computing for high resolution remote sensing image*. Hangzhou: Zhejiang University.

Kathleen, Hornsby and Max, J. Egenhofer, 2000. Identity-based change: a foundation for spatio-temporal knowledge representation. *International Journal of Geographical Information Science*, 14 (3), 207–224.

Karun, A.K. and Chitharanjan, K., 2013. A review on Hadoop–HDFS infrastructure extensions. *In*: *2013 IEEE conference on information & communication technologies*, 11–12 April 2013 Thuckalay. New York: IEEE, 132–137.

Lei, S.M., 2010. Research on the key techniques of massive image data management based on Hadoop. Thesis (Master). National University of Defense Technology.

Liu, X. F., *et al.*, 2003. Method for basin palaeotectonic reconstruction based on GIS. *Editorial Board of Geomatics and Information Science of Wuhan University*, 28 (2), 197–201+207.

Liu, X.H., Wu, X.C., and Luo, X.G., 2013. Object-oriented geological disaster data model and spatio-temporal process expression. *Geomatics and Information Science of Wuhan University*, 38 (8), 958–961.

Long, Y., *et al.*, 2011. A multi-agent model for urban form, transportation energy consumption and environmental impact integrated simulation. *Acta Geographica Sinica*, 66 (8), 1033–1044.

Ma, Y., *et al.*, 2015. Towards building a data-intensive index for big data computing – A case study of Remote Sensing data processing. *Information Sciences*, 319 (C), 171–188.

Papailiou, N., *et al.*, 2014. H2RDF+: an efficient data management system for big RDF graphs. *In*: *SIGMOD '14 proceedings of the 2014 ACM SIGMOD international conference on management of data*, 22–27 June 2014 Snowbird. New York: ACM, 909–912.

Ranjan, R., *et al.*, 2016. Advances in methods and techniques for processing streaming big data in datacentre clouds. *IEEE Transactions on Emerging Topics in Computing*, 4 (2), 262–265.

Ren, X.X., 2011. Research of job scheduling based on Hadoop platform. Thesis (Master). Tianjin Normal University.

Shin, D. H., and Kim, W. Y., 2008. Applying the technology acceptance model and flow theory to cyworld user behavior: implication of the web2.0 user acceptance. *Cyberpsychology & Behavior : The Impact of the Internet, Multimedia and Virtual Reality on Behavior and Society*, 11 (3), 378.

Wang, L., Chen, B., and Liu, Y., 2013. Distributed storage and index of vector spatial data based on HBase. *In*: *2013 21st international conference on geoinformatics*, 20–22 June 2013 Kaifeng. New York: IEEE, 1–5.

Wang, L., *et al.*, 2014. IK-SVD: dictionary learning for spatial big data via incremental atom update. *Computing in Science & Engineering*, 16 (4), 41–52.

Wang, L., *et al.*, 2015. Particle swarm optimization based dictionary learning for remote sensing big data. *Knowledge-Based Systems*, 79 (C), 43–50.

White, T. and Cutting, D., 2009. Hadoop : the definitive guide. *O'reilly Media Inc Gravenstein Highway North*, 215 (11), 1–4.

Wu, C. and Lue, G., 2008. Improved event-process based on spatiotemporal model. *Journal of wuhan university*, 33 (12), 1250–1253+1278.

Wu, C.Q., Ren, P.G., and Wang, X.F., 2015. Survey on semantic-based organization and search technologies for network big data. *Chinese Journal of Computers*, 38 (1), 1–17.

Zhang, R., Deren, L., and Song, D., 2002. An object-oriented spatio-temporal data model. *Acta Geodaetica Et Cartographic Sinica*, 31 (1), 87–92.

Zhou, L.Z. and Chen, Q.K., 2012. HBase-based storage system for wireless sensor information of agriculture. *Computer Systems & Applications*, 32 (7), 1920–1923.

Zhu, X.L. and Zhai, Z.G., 2013. A mass image storage technology based on HBase.*China CIO News*, 8, 22–24.