



Article

# LandQ<sup>v2</sup>: A MapReduce-Based System for Processing Arable Land Quality Big Data

Xiaochuang Yao <sup>1,\*</sup> , Mohamed F. Mokbel <sup>2</sup>, Sijing Ye <sup>3</sup>, Guoqing Li <sup>1</sup>, Louai Alarabi <sup>2</sup>, Ahmed Eldawy <sup>4</sup>, Zuliang Zhao <sup>5</sup> , Long Zhao <sup>5</sup> and Dehai Zhu <sup>5,\*</sup>

<sup>1</sup> Institute of Remote Sensing and Digital Earth, Chinese Academy of Sciences, Beijing 100094, China; ligq@radi.ac.cn

<sup>2</sup> Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN 55455, USA; mokbel@umn.edu (M.F.M.); louai@cs.umn.edu (L.A.)

<sup>3</sup> State Key Laboratory of Earth Surface Processes and Resource Ecology, Beijing Normal University, Beijing 100875, China; yesj@bnu.edu.cn

<sup>4</sup> Department of Computer Science and Engineering, University of California, Riverside, CA 92521, USA; eldawy@cs.ucr.edu

<sup>5</sup> College of Information and Electrical Engineering, China Agricultural University, Beijing 100083, China; zlzhao@cau.edu.cn (Z.Z.); zhaolcau@163.com (L.Z.)

\* Correspondence: yaocx@radi.ac.cn (X.Y.); zhudehai@263.net (D.Z.); Tel.: +86-1881-135-6282 (X.Y.)

Received: 25 May 2018; Accepted: 6 July 2018; Published: 10 July 2018



**Abstract:** Arable land quality (ALQ) data are a foundational resource for national food security. With the rapid development of spatial information technologies, the annual acquisition and update of ALQ data covering the country have become more accurate and faster. ALQ data are mainly vector-based spatial big data in the ESRI (Environmental Systems Research Institute) shapefile format. Although the shapefile is the most common GIS vector data format, unfortunately, the usage of ALQ data is very constrained due to its massive size and the limited capabilities of traditional applications. To tackle the above issues, this paper introduces LandQ<sup>v2</sup>, which is a MapReduce-based parallel processing system for ALQ big data. The core content of LandQ<sup>v2</sup> is composed of four key technologies including data preprocessing, the distributed R-tree index, the spatial range query, and the map tile pyramid model-based visualization. According to the functions in LandQ<sup>v2</sup>, firstly, ALQ big data are transformed by a MapReduce-based parallel algorithm from the ESRI Shapefile format to the GeoCSV file format in HDFS (Hadoop Distributed File System), and then, the spatial coding-based partition and R-tree index are executed for the spatial range query operation. In addition, the visualization of ALQ big data with a GIS (Geographic Information System) web API (Application Programming Interface) uses the MapReduce program to generate a single image or pyramid tiles for big data display. Finally, a set of experiments running on a live system deployed on a cluster of machines shows the efficiency and scalability of the proposed system. All of these functions supported by LandQ<sup>v2</sup> are integrated into SpatialHadoop, and it is also able to efficiently support any other distributed spatial big data systems.

**Keywords:** spatial big data; parallel processing; MapReduce; arable land quality (ALQ); GIS

## 1. Introduction

Arable land is a foundational resource for national food security. Arable land quality (ALQ) data are about the quality evaluation of arable land by classification and gradation. With the rapid development of spatial information technologies, the annual acquisition and update of ALQ data covering the country have become more accurate and faster. ALQ data are dominated by spatial

vector data in the ESRI shapefile format. In China, the government began to perform all round arable land quality monitoring [1,2] and evaluation work in 2011. In addition, in 2013, the government formed a relatively complete and large arable land quality dataset of approximately 2.51 TB in the ESRI shapefile format [3]. To make such an enormous amount of data readily available requires a management information system with high-performance computing that will play a crucial role in offering accessibility for the government to make scientific decisions and ensuring valuable spatial data safety, readily, and practically.

The traditional standalone version GIS development pattern has been challenged by the volume, velocity, and variety of the arable land quality big dataset. Our team has developed a GIS cluster-based management information system, LandQ<sup>v1</sup> (version 1) [3]. In the first version, the ArcGIS platform was used as the basic development framework and Oracle 11g was adopted as the spatial database. However, along with the continuous application of the system (LandQ<sup>v1</sup>), the increasing challenges and overhead make it very hard to use such a rich spatial vector dataset from the national scale. On the one hand, data preprocessing takes a large amount of time and is not ideal in terms of system performance. On the other hand, a large number of tables not only cause data organization difficulties, but also bring about client service scheduling problems. In addition, there are serious deficiencies in the data querying and visualizations for ALQ big data.

The rise of big data has brought about a remarkable change in the traditional GIS industry, especially based on cloud computing technology [4,5], which has provided a potential solution for addressing these computational challenges. Several processing and analyzing platforms have been developed for spatial big data, such as Hadoop-GIS [6], SpatialHadoop [7], GeoMesa [8], GeoSpark [9], ST-Hadoop [10], and others. Meanwhile, related application systems have also sprung up from different fields around the world, including satellite or remote sensing data [11,12], sensor data [13], social media [14], and others [15,16]. For the various application areas, the directions and emphasis of the above systems focus on different topics. In this paper, we take the national land and resource information fields, especially the arable land quality (ALQ) data, into consideration. For the ALQ big data, cloud computing technology is adopted to solve the practical problems in data storage and management.

LandQ<sup>v2</sup> (version 2) is a MapReduce-based parallel processing system that lays out the necessary infrastructure to store, index, query, and visualize the nation's arable land quality (ALQ) big data. To solve the problem of data storage in the cloud environment, we designed a GeoCSV model to support the data processing, and then, we used a spatial coding-based approach to partition the whole dataset and build a distributed R-tree index in HDFS. In addition, the spatial query and visualization operations are developed and implemented within the MapReduce program. All these functions, supported by LandQ<sup>v2</sup>, are integrated into SpatialHadoop [7], which is a very successful framework for spatial big data, and it is also able to efficiently support any other distributed big spatial data systems.

## 2. LandQ<sup>v2</sup> System Overview

Figure 1 shows the LandQ<sup>v2</sup> system overview. It was developed in a cloud computing environment, which mainly includes four levels, namely, the spatial data infrastructure layer, the cloud computing platform layer, the key technologies layer, and the system application layer. The spatial data infrastructure layer is composed of ordinary servers that form a resilient and scalable cluster, mainly providing data storage and computing resources for the upper layer. In the cloud computing platform layer, this paper selects the mainstream Hadoop ecosystem, and the current system architecture mainly involves the processing system (MapReduce), program language (Pig), and storage system (HDFS). The core content of this system is the key technologies layer, which includes the vector data cloud storage model, spatial data partitioning, the distributed R-tree index, the tile pyramid model-based visualization, and others, as shown in Figure 1. The system application layer is mainly based on the above layers to handle and leverage arable land quality (ALQ) big data.

In the system development process, in order to improve the compatibility and continuity of LandQ<sup>v2</sup>, the existing open source standards and basic tools are referenced. For example, the cloud storage model—GeoCSV—is proposed based on the OGC (Open Geospatial Consortium) standard, characteristics of the distributed storage and parallel programming model. For spatial data processing, the ESRI Geometry API (Application Programming Interface) for Java is used to complete the analysis of simple geometric object elements. In addition, in the data visualization engine, the mainstream ArcGIS Runtime for the WPF (Windows Presentation Foundation) desktop development and ESRI Web API for web development are used to meet the application requirements of the system client. All of these functions supported by LandQ<sup>v2</sup> are integrated into SpatialHadoop.

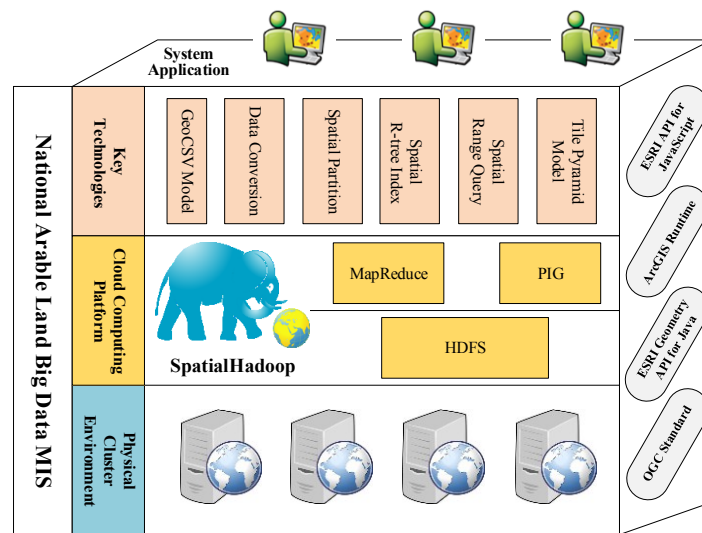


Figure 1. The LandQ<sup>v2</sup> system overview.

### 3. Key Technologies in LandQ<sup>v2</sup>

#### 3.1. Data Preprocessing

Arable land quality (ALQ) data are mainly vector-based spatial data in the ESRI Shapefile format, which is a mainstream spatial vector data storage format [17] because of its simple structure, high precision, and rapid display capabilities. However, in the era of spatial big data, this format faces severe challenges. On the one hand, its storage volume is constrained by the 2 GB upper limit, which will result in thousands of small files and will increase the difficulty of data management and processing. On the other hand, there are obvious shortcomings in the field types, index efficiency, and network transmission. In addition, the visualization of the Shapefile generally requires professional GIS tools or software to operate. To solve the above shortcomings, GeoCSV is proposed based on the OGC standard. This format also offers to the idea of distributed storage and the parallel programming model.

##### 3.1.1. Vector Data Cloud Storage Model

As Figure 2 shown, the data structure of GeoCSV [18] adopts the object-based vector data model to store spatial geometric elements in the cloud environment. GeoCSV uses the simple Key-Value storage model and the OGC-WKT format [19] to describe the spatial geometry. It takes the advantages of CSV (comma-separated values) files to organize the spatial vector data, that is, each record represents only one spatial geometry object. CSV file contains a number of data records. This is in line with the cloud computing platform, which is conducive to spatial data segmentation, processing, and analysis.

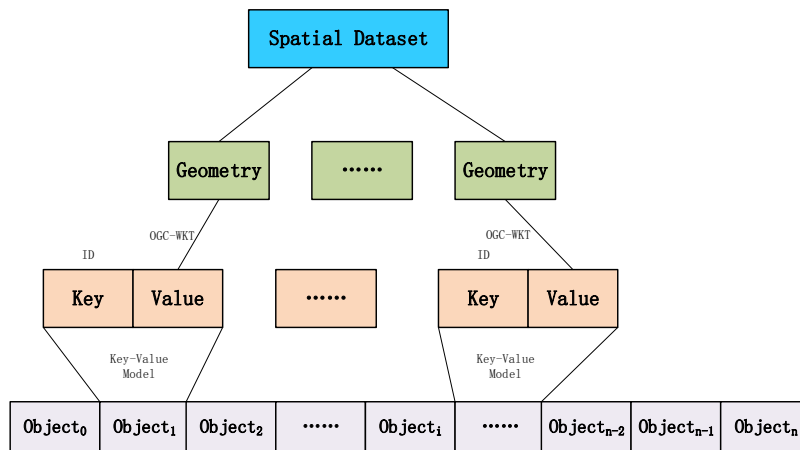


Figure 2. The spatial vector data cloud storage model, GeoCSV [18].

Based on the GeoCSV model, this model has the following advantages: (1) object-oriented storage of existing spatial vector data, and the parallel computation of fine granularity is supported; (2) simple data structure with OGC standard data and the advantages of network transmission; (3) CSV is used for the separation of storage, which is conducive to the distributed processing of data segmentation, processing, and analysis; and (4) spatial data and attribute data formats are unified, and it is easy to add a field or adjust the data structure.

### 3.1.2. Data Conversion

Data conversion from Shapefile to GeoCSV is executed by one MapReduce job as shown in Figure 3. In this way, batch conversions can be implemented, which will change multiple small Shapefiles into one large GeoCSV file. Since a Shapefile is composed of multiple sub-files (.shp, .shx, .dbf, etc.), and each sub-file contains a unique header file, the conversion of a single Shapefile file will be treated as a separate subtask in the implementation of the data conversion algorithm. The task is done primarily as a two-part process, namely, Shapefile parsing and spatial object reconstruction in GeoCSV.

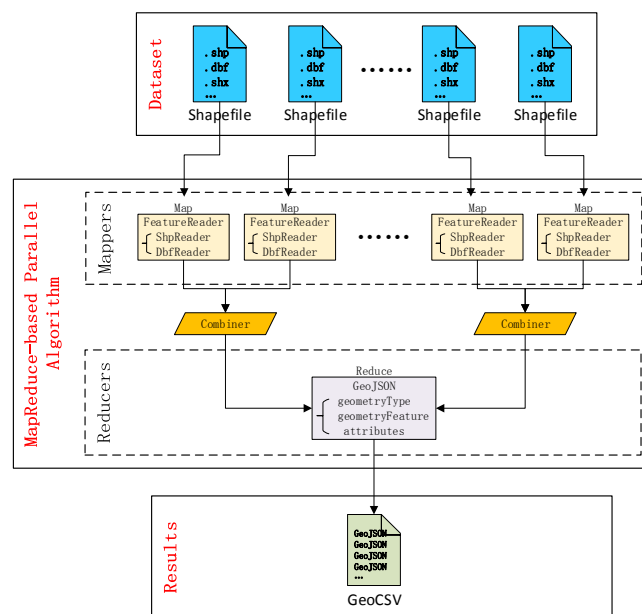


Figure 3. The MapReduce-based parallel data conversion algorithm.

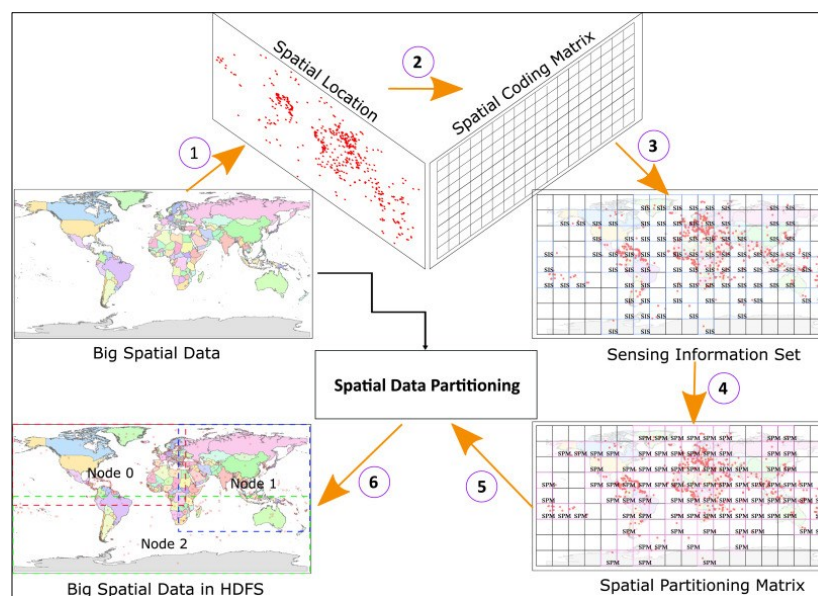


### 3.2. Spatial Index

In the distributed storage environment, a spatial dataset will be separated into several blocks dispensed to different nodes in the cluster [20], and the spatial index technology based on a single node cannot be used directly. Compared with the centralized index, the distributed index increases the communication protocol and communication overhead among the nodes in the cluster [21]. The index mechanism of a distributed storage system must take full account of the distributed systems and the organization of the data storage.

#### 3.2.1. Data Partitioning

In this section, we present the spatial partitioning architecture based on the spatial coding-based approach (SCA) [22]. Figure 4 summarizes the six steps, which can be done by two MapReduce jobs. (1) Computing the spatial location: the central point is used instead of the spatial objects; (2) Defining the spatial coding matrix (SCM): we define the spatial coding values as a spatial coding matrix; (3) Computing the sensing information set (SIS): the SIS includes the spatial code, location, size, and count; (4) Computing the spatial partitioning matrix (SPM): SPM is built according to the SIS and the blocked size in HDFS; (5) Spatial data partitioning: every spatial object will be matched with the spatial coding and written into the data blocks in HDFS; (6) Allocating the data blocks: all blocks will be distributed into the nodes in the cluster with their spatial codes.



**Figure 4.** The algorithm of spatial partitioning based on SCA (Spatial Coding-based Approach) [22].

#### 3.2.2. Distributed R-Tree Index

The R-tree index is the most influential data access method in the GIS database [23]. As Figure 5 shows, the distributed R-tree index includes the local index and global index [24]. The local index, also called a data block index, is created for the collection of spatial object information in the data block after the spatial partitioning steps in HDFS. Here, this index will generate an index header file for each data block in HDFS, and the contents of the index header file include the envelope rectangle (MBR), the count of spatial objects (Count), offset to the start (Offset), and other basic information. The global index is created from the local index. The purpose of this phase is to build the global index structure, indexing all partitions in HDFS. By concatenating all the local index files into one file, the global index file will be generated. In addition, the content of the global index file includes the data block id (ID), the envelope rectangle (MBR), the total size (Size), and others.

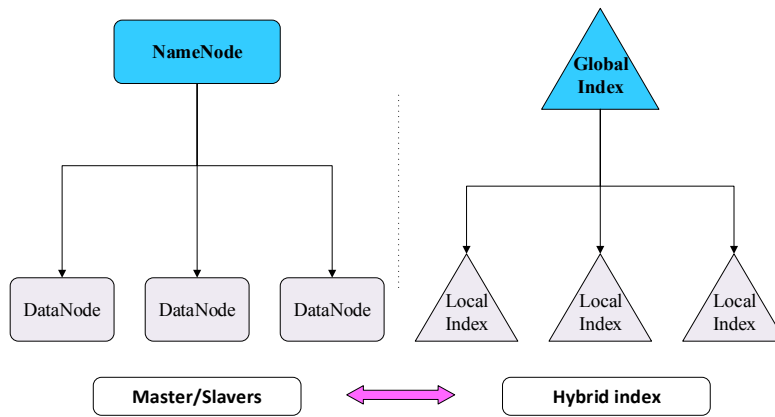


Figure 5. The distributed R-tree index.

### 3.3. Spatial Range Query

The spatial query operation is closely related to the spatial data storage model and spatial index, and it can be regarded as their reverse process. In the non-indexed Hadoop cloud environment, a spatial query needs to traverse all the spatial information records to match the corresponding results, and for large-scale spatial data, the query efficiency is extremely low.

As shown in Figure 6, this paper combines the spatial range query algorithm into three phases: (a) the Global filtering phase. The content stored in the global index is the basic information of all the data blocks in HDFS. The size of the global index file is relatively small, so this phase is executed at the master node. All query operations need to go through the global filtering phase first and the global index file usually resides in the memory of the master node; (b) the Local filtering stage. The local filtering phase is mainly for filtering the candidate sets in the data blocks from the global filtering results. The local index file is mainly the leaf node of the distributed R-tree stored in the data block header file; (c) the Refining stage. Through step (b), it can get all the leaf nodes in the current data block that intersects the query range. The refining stage mainly matches each spatial geometric object in the candidate set; if the object intersects or is included in the spatial query range, it will be put in the results file, otherwise, it will not perform this operation.

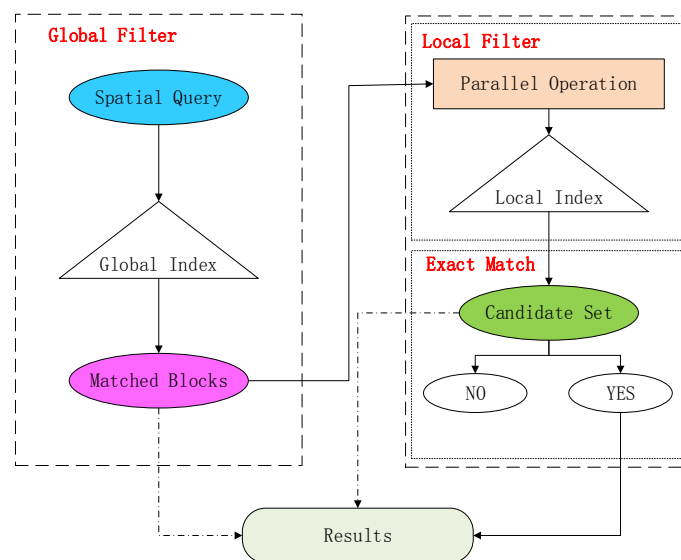


Figure 6. The spatial query steps based on the distributed R-tree index.

### 3.4. Data Visualization

The tile pyramid model uses “vertical grading, horizontal block” thinking, and the map data are divided into uniform rectangular tiles where the number of levels depends on the map scale, and the tile number is decided by the picture size [25].

The visualization based on the tile pyramid model is developed by employing a three phase technique [26]: (1) The partitioning phase: here, we use the default Hadoop partitioning to split the total input spatial big data into partitions in HDFS; (2) The plotting phase: through the internal loop, all tiles occupied by each spatial object will be plotted in the tile pyramid model; (3) The merging phase: the partial images having the same index code will be combined into one final image. As shown in Figure 7, each tile in the map pyramid model is indexed according to the level, row, and column as the unique identifier. In the parallel algorithm, the construction of each tile can be regarded as a separate sub-task, and the entire tile construction of the pyramid model can be regarded as a collection of multiple subtasks.

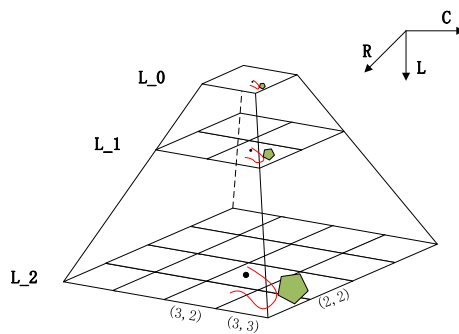


Figure 7. The schematic diagram of the tile pyramid model.

In general, we assume that the enclosing rectangle of the space object shape is  $mbr$ , and the maximum and minimum coordinates are  $(x1, y1)$  and  $(x2, y2)$ , respectively. Then, the following formula is used to get the spatial object in the pyramid model. The cover of the map tile ( $r$ ) column ( $c$ ) is

$$r_{min}(\text{shape}) = \left\lceil \frac{mbr_{y1} - MBR_{y1}}{MBR_{length}/2^l} \right\rceil \quad (1)$$

$$r_{max}(\text{shape}) = \left\lceil \frac{mbr_{y2} - MBR_{y1}}{MBR_{length}/2^l} \right\rceil \quad (2)$$

$$c_{min}(\text{shape}) = \left\lceil \frac{mbr_{x1} - MBR_{x1}}{MBR_{width}/2^l} \right\rceil \quad (3)$$

$$c_{max}(\text{shape}) = \left\lceil \frac{mbr_{x2} - MBR_{x1}}{MBR_{width}/2^l} \right\rceil \quad (4)$$

where the  $[r_{min}, r_{max}]$  and  $[c_{min}, c_{max}]$  for the space object shape of the map tile correspond to the line number and column number interval, respectively, and MBR is for the entire map. The  $l$  is the pyramid level.

In particular, if the map tile size is  $p$  and the level has a spatial resolution of  $r$ , the spatial resolution of the tile in the  $x$  and  $y$  directions satisfies the following conditions:

$$r_x = \frac{MBR_{width}}{p \times 2^l} \quad (5)$$

$$r_y = \frac{MBR_{length}}{p \times 2^l} \quad (6)$$

#### 4. System Test and Results

In this section, the key technologies mentioned above are tested with the national arable land quality (ALQ) big data. The content of the test in this paper includes three aspects: data conversion, spatial query operation, and visualization performance. The test results will be given bellow.

##### 4.1. Data Conversion

The parallel conversion experiment of the vector data mainly includes two aspects. First, the parallel algorithm proposed in this paper is compared with the data transformation algorithm in the ArcGIS software. On the other hand, the parallel algorithm is tested in the cluster environment. The experimental environment in this paper includes the stand-alone and cluster environment. Table 1 configures the information for the experimental environment. Table 2 shows the test data and that the size of the ESRI Shapefile varies from 2 GB to 128 GB.

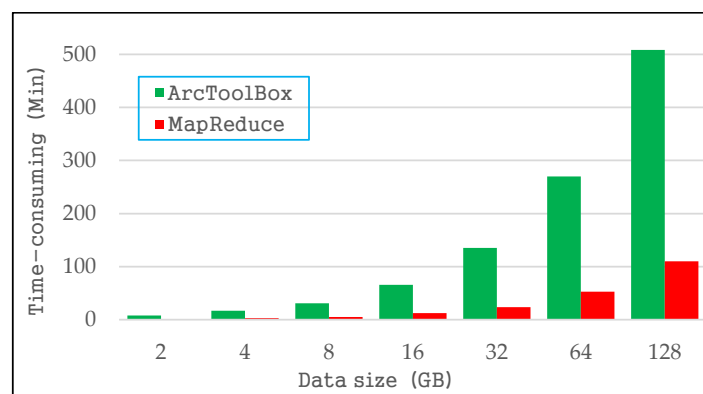
**Table 1.** The test environment for the data conversion algorithm.

Test Environment	System Configuration
Stand-alone	Windows 10, 64 GB Memory, 600 GB Hard disk ArcGIS 10.2
	Ubuntu 15, 64 GB Memory, 600 GB Hard disk Hadoop 2.7
Cluster	Ubuntu 15, 24 GB Memory, 2 TB Hard disk Hadoop 2.7

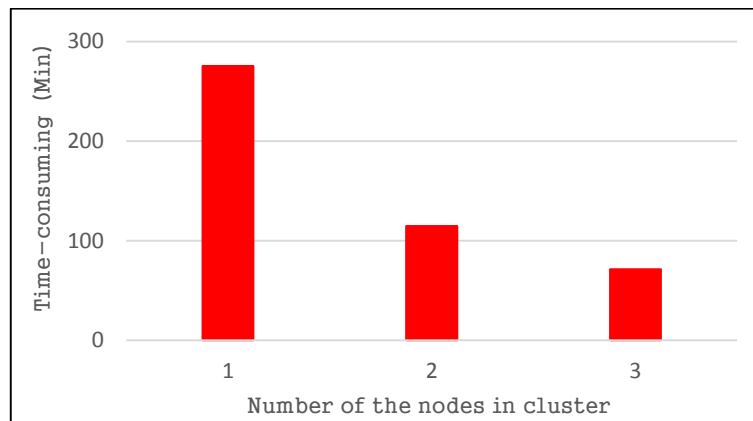
**Table 2.** The datasets for the parallel conversion test.

Size/GB	Shapefile Number	Feature Number
2	86	1,361,127
4	257	2,512,413
8	304	5,218,028
16	680	11,077,580
32	1120	21,615,328
64	3008	43,847,424
128	5561	85,558,229

The experimental results are shown in Figures 8 and 9. It can be seen that the conversion time of the two algorithms increases with the growth of the size of the ESRI Shapefile. At the same time, it is fairly clear that the parallel conversion algorithm based on the MapReduce proposed in this paper is more efficient in terms of time. Moreover, with the increase of the number of ESRI Shapefile, the efficiency growth is clearer, approximately 4 to 6 times to the ArcToolBox.



**Figure 8.** The time-consuming comparison of data conversion in a stand-alone environment.



**Figure 9.** The time-consuming comparison of data conversion in a cluster environment.

Figure 9 shows the time consuming comparison of the data conversion efficiency in the cluster environment. Here, we only test a data volume of 64 GB. Through the test results, we can find that the execution time of the algorithm becomes shorter with the increase of the number of cluster nodes, which fully embodies the advantages of the cloud computing environment.

## 4.2. Data Query

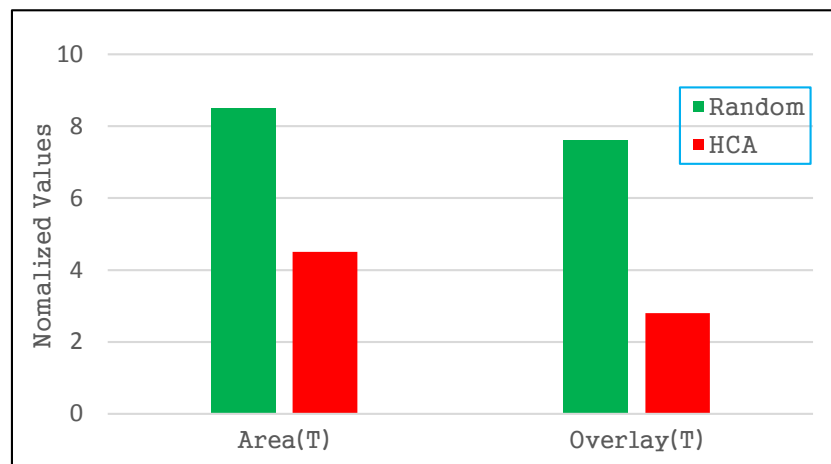
### 4.2.1. Index Quality of the R-Tree

The R-tree spatial index quality can be measured by the envelope rectangle of its index tree hierarchies. The calculation formula is as follows, where  $Area(T)$  in Equation (7) represents the total area covered by the MBR of each index layer of the R-tree. Generally, the less the coverage area is, the better the R-tree index performance. In Equation (8),  $Overlap(T)$  represents the overlap area between the MBRs of each index layer in the R-tree. The larger the overlap area is, the more space the query path will have, which leads to a greater query time consumption and greatly reduces the efficiency of the data retrieval. Thus, for overlapping areas, the smaller the area is, the better the R-tree performance will be. The R-tree spatial index quality analysis formula is as follows:

$$Area(T) = \sum_{i=1}^n (T_i MBR) \quad (7)$$

$$Overlap(T) = \sum_{i=1}^n \sum_{j=i+1}^n Area(T_i MBR \cap T_j MBR) \quad (8)$$

As shown in Figure 10, spatial coding-based data partition (Hilbert coding-based approach, HCA) shows a better index quality than random sampling for building an R-tree in both  $Area(T)$  and  $Overlap(T)$ . The main reasons include the following two aspects. (1) Due to the randomness of the sample set and lost spatial features. The data partitioning method based spatial-coding can make good use of the adjacent features of spatial coding, which greatly guarantees the spatial distribution of the vector data; (2) the data partitioning method based on spatial sampling. Only the spatial location information of the sample is considered, however, the other one is considered with more influencing factors, including spatial location information, the number of elements, the amount of data bytes, and so on. In addition, the sampling based method is also due to the randomness of the samples, the partitioning strategy for the same dataset and the same sampling rate are different, and the algorithm lacks stability.



**Figure 10.** The index quality comparison of the R-tree based on random and HCA.

#### 4.2.2. Spatial Query Performance

Based on the distributed R-tree, here, we test the query performance of LandQ<sup>v2</sup>. We measure the query time consumption in the different numbers of jobs and nodes.

In experiment 1, the result in Figure 11 shows the query performance comparison between the different job numbers with two modes, namely, the non-index and R-tree index. It can be seen that the spatial query execution time for the data increases with the increase of the spatial parallel access times, and there is a linear trend of growth for the non-index model, which is mainly due to traversing all the data blocks regardless of any spatial query. For the R-tree index model, because the search is only executed with the relevant data block, there is no obvious linear growth trend and there is a great relationship with the spatial query itself. In addition, for both models, the more query jobs are queried, the longer it will take.



**Figure 11.** The query performance comparison between the different job numbers.

Experiment 2 tests the vector data query efficiencies with spatial index in different cluster nodes. The experimental results are shown in Figure 12. It can be seen that the efficiency of the data query increases with the increase in the number of cluster nodes. At the same time, the parallel query algorithm in this paper can perform well and shows good superiority.



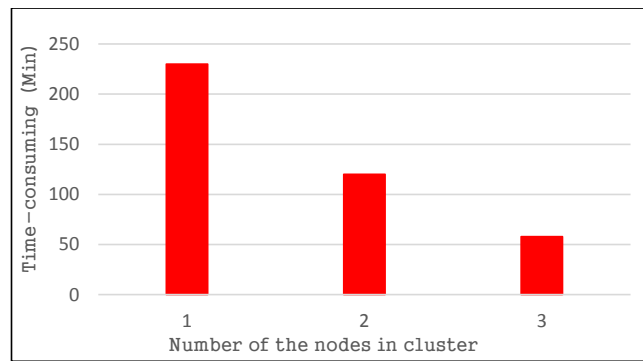


Figure 12. The query performance comparison between the different node numbers.

Through the above comparison tests, the following conclusions can be drawn. (1) The efficiency of the spatial query based on the spatial index has been obviously improved; (2) The R-tree index based on the spatial coding-data partitioning also showed a good advantage in the spatial query operation; (3) With the increase of the nodes, the efficiency of the spatial parallel query can be significantly improved, and it can expand the number of clusters to improve the efficiency of the data retrieval.

4.3. Visualization Performance

4.3.1. Tile Pyramid Construction Performance

In this section, we test two experiments. One experiment is a comparison between the parallel construction algorithm proposed in this paper and the existing mainstream ArcGIS server. The other tests the algorithm performance in the cluster. The experimental data is in the Table 3.

Table 3. The datasets for visualization.

Dataset	Size	Shapefile Count	Feature Count
Province	3.4 GB	31	359,227
County	154.5 GB	2673	63,033,494

Experiment 1 was conducted in the same hardware environment and the tile pyramid model parameters were consistent. The test results are shown in Figure 13. Under the same test environment, the algorithm proposed in this paper is better. In experiment 2, the efficiency of the parallel construction of different datasets is compared with the number of different cluster nodes. The tile pyramid level is set to level 8. The result is shown in Figure 14. Under the same task, the greater the number of cluster nodes was, the clearer the advantages of tile pyramid parallel construction were.

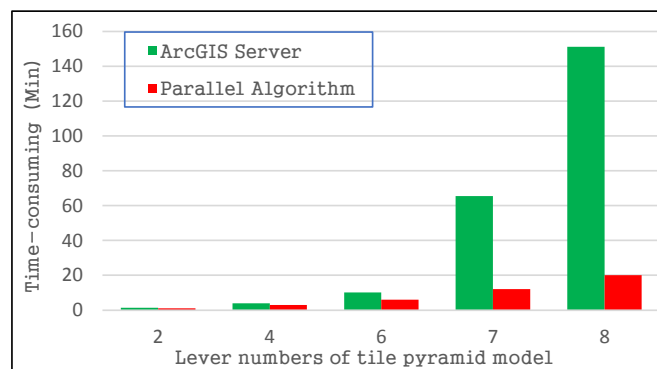


Figure 13. The time-consuming comparison of the tile pyramid construction in a single machine environment.

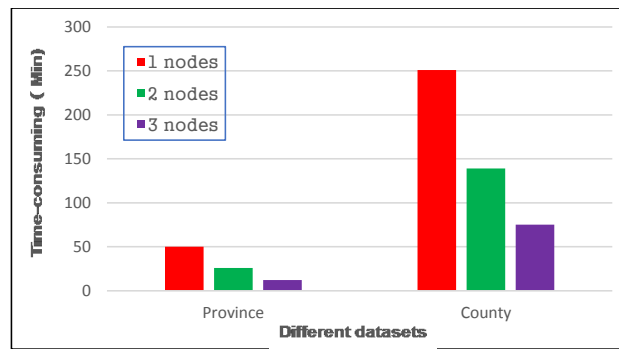


Figure 14. The time-consuming comparison of the tile pyramid construction in a cluster environment.

#### 4.3.2. Visualization for Arable Land Quality Big Data

Based on the parallel construction algorithm of the tile pyramid model, the provincial and county level of arable land quality (ALQ) big data are respectively visualized. Following the OGC map standard, we construct the map tiles and integrate them with the ESRI API for JavaScript very well.

For the 2013 national and provincial level of arable land quality, there were equal data map elements of the tile pyramid parallel construction. In this paper, we showed the national economic classification, which reflects the economic value of the arable land. The resulting tile details are shown in Table 4. The tile pyramid model has 8 levels of map scale ranging from 1:20 million to 1:31.25 million. The county dataset had a parallel build time of 1 h 15 min 7 s, and the provincial data had a parallel build time of 11 min and 52 s.

Table 4. The datasets for the parallel conversion test.

Level	Scale (million)	Tile Number (County)	Size (County)	Tile Number (Province)	Size (Province)
0	1:4000	6	35.1 KB	6	36.4 KB
1	1:2000	15	103 KB	15	100 KB
2	1:1000	44	331 KB	44	504 KB
3	1:500	124	1.06 MB	123	837 KB
4	1:250	390	3.71 MB	380	2.41 MB
5	1:125	1312	12.3 MB	1265	6.6 MB
6	1:62.5	4523	36.9 MB	4311	16.6 MB
7	1:31.25	15,138	111 MB	15,183	40.1 MB

Figures 15 and 16 show the national arable land quality data global browsing and local browsing diagrams, respectively. In line with the WMS standard, the resulting map tiles are stored in folders with different scales which will be accessed and loaded by the Web client. At the same time, it can integrate well with other layers, such as national administrative boundary, railway, water, and other layers in the client.

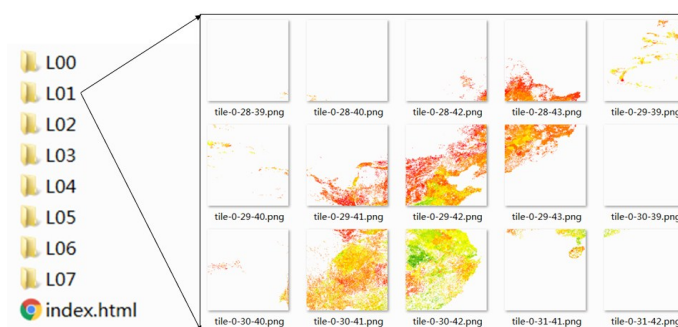


Figure 15. The map tiles of the national arable land quality dataset.



**Figure 16.** The schematic diagram of visualization for the national arable land quality dataset.

## 5. Discussion

The increased amount of the arable land quality (ALQ) data has actually brought challenges for processing and managing the vector-based spatial big data, especially in terms of data processing efficiency. Traditional GIS solutions have been unable to meet the needs of the national application. Here, we present a MapReduce-based parallel processing system, LandQ<sup>v2</sup>, for national arable land quality (ALQ) big data. LandQ<sup>v2</sup> is a complete set of application systems and is integrated into SpatialHadoop. Similarly, the relevant technologies can also be applied to other areas.

In LandQ<sup>v2</sup>, four key technologies are designed and implemented. It is based on the cloud storage model of the spatial vector data (GeoCSV). The many shortcomings of the ESRI Shapefile will be addressed and solved in the data storage and process. The batch conversion algorithm with MapReduce improves the efficiency of data processing from the ESRI Shapefile to GeoCSV. Aimed at the problem of data query efficiency, a spatial coding-based approach for partitioning the spatial big data is presented, and then the distributed R-tree index is built for the spatial range query. This method will improve the query performance of the spatial big data, as well as the data balance in HDFS [22]. For the visualization of the remote sensing data, the tile pyramid model is a good and mature solution [27,28]. In this paper, we use this model to solve the problem of visualizing the vector-based spatial big data. The tile pyramid model-building algorithm is proposed and parallelized with the MapReduce program.

We tested the above key technologies and system functions. By comparing these features with existing GIS tools, all of the parallel algorithms, including data conversion, data partitioning, distributed R-tree index, data query, and visualization, show good performance and, at the same time, they can take full advantages of the cluster scalability. Compared with LandQ<sup>v1</sup>, the visualization of the ALQ dataset in national level makes the macroscopic grasp of data more comprehensive and scientific, which will be useful to policymakers.

## 6. Conclusions

Recent advancements in cloud computing technology have offered a potential solution to the big data challenges [29]. This paper presents LandQ<sup>v2</sup>, which is a MapReduce-based parallel processing system for arable land quality (ALQ) big data that uses the Hadoop cloud computing platform. The contents of system architecture, key technologies, and the tests shown in this study are efficient enough to bring the following remarkable advantages of the developed system: (1) The spatial vector big data cloud storage model, GeoCSV, which is based on the characteristics of spatial vector data

and the advantages of the Hadoop cloud platform; (2) the distributed R-tree index in LandQ<sup>v2</sup>, the data partitioning strategy based on spatial coding is designed, and the parallel construction of the distributed R-tree index is realized. Through experiments, the efficiency and feasibility of different distributed spatial indexing algorithms are verified from two perspectives: spatial index quality and data balance; (3) parallel processing for ALQ big data. This paper carries out parallel processing methods including a data conversion algorithm, spatial range query, and tile pyramid model construction. All these parallel algorithms are implemented with the MapReduce program. In addition, using ALQ big data, the experiments are used to verify the efficiency and feasibility of the proposed vector data processing algorithm.

**Author Contributions:** X.Y. developed the LandQ<sup>v2</sup> and wrote this paper. M.F.M., L.A. and A.E. are the contributors of the SpatialHadoop and provided key technical guidance for the system development. S.Y., Z.Z., and L.Z. assisted in the system development. G.L. and D.Z. are the supporters of this project and supervised this paper.

**Funding:** This work was sponsored by National Key Research and Development Program of China from MOST (2016YFB0501503).

**Acknowledgments:** We would like to thank the anonymous reviewers and editors for commenting on this paper.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Yao, X.; Zhu, D.; Ye, S.; Yun, W.; Zhang, N.; Li, L. A field survey system for land consolidation based on 3S and speech recognition technology. *Comput. Electron. Agric.* **2016**, *127*, 659–668. [[CrossRef](#)]
2. Ye, S.; Zhu, D.; Yao, X.; Zhang, N.; Fang, S.; Li, L. Development of a highly flexible mobile GIS-based system for collecting arable land quality data. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2014**, *7*, 4432–4441. [[CrossRef](#)]
3. Yao, X.; Yang, J.; Li, L.; Yun, W.; Zhao, Z.; Ye, S.; Zhu, D. LandQ<sup>v1</sup>: A GIS cluster-based management information system for arable land quality big data. In Proceedings of the 6th International Conference on Agro-Geoinformatics (Agro-Geoinformatics), Fairfax, VA, USA, 7–10 August 2017; pp. 1–6.
4. Huang, Q.Y.; Yang, C.W.; Liu, K.; Xia, J.Z.; Xu, C.; Li, J.; Gui, Z.P.; Sun, M.; Li, Z.L. Evaluating open-source cloud computing solutions for geosciences. *Comput. Geosci.* **2013**, *59*, 41–52. [[CrossRef](#)]
5. Li, Z.; Yang, C.; Liu, K.; Hu, F.; Jin, B. Automatic scaling Hadoop in the cloud for efficient process of big geospatial data. *ISPRS Int. Geo-Inf.* **2016**, *5*, 173. [[CrossRef](#)]
6. Aji, A.; Sun, X.; Vo, H.; Liu, Q.; Lee, R.; Zhang, X.; Saltz, J.; Wang, F. Demonstration of Hadoop-GIS: A spatial data warehousing system over mapreduce. In Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, Orlando, FL, USA, 5–8 November 2013; ACM: New York, NY, USA, 2013; pp. 528–531.
7. Eldawy, A.; Mokbel, M.F. A demonstration of spatialhadoop: An efficient mapreduce framework for spatial data. *Proc. VLDB Endow.* **2013**, *6*, 1230–1233. [[CrossRef](#)]
8. Hughes, J.N.; Annex, A.; Eichelberger, C.N.; Fox, A.; Hulbert, A.; Ronquest, M. Geomesa: A distributed architecture for spatio-temporal fusion. In Proceedings of the Geospatial Informatics, Fusion, and Motion Video Analytics V, Baltimore, MD, USA, 20–21 April 2015; p. 94730F.
9. Yu, J.; Wu, J.; Sarwat, M. Geospark: A cluster computing framework for processing large-scale spatial data. In Proceedings of the 23rd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, Seattle, WA, USA, 3–6 November 2015; ACM: New York, NY, USA, 2015; pp. 1–4.
10. Alarabi, L. St-Hadoop: A mapreduce framework for big spatio-temporal data. In Proceedings of the ACM International Conference on Management of Data, Chicago, IL, USA, 14–19 May 2017; ACM: New York, NY, USA, 2017; pp. 40–42.
11. Mueller, N.; Lewis, A.; Roberts, D.; Ring, S.; Melrose, R.; Sixsmith, J.; Lymburner, L.; McIntyre, A.; Tan, P.; Curnow, S.; et al. Water observations from space: Mapping surface water from 25 years of landsat imagery across Australia. *Remote Sens. Environ.* **2016**, *174*, 341–352. [[CrossRef](#)]
12. Li, J.; Meng, L.; Wang, F.Z.; Zhang, W.; Cai, Y. A map-reduce-enabled solap cube for large-scale remotely sensed data aggregation. *Comput. Geosci.* **2014**, *70*, 110–119. [[CrossRef](#)]

13. Zhong, Y.; Fang, J.; Zhao, X. Vegaindexer: A distributed composite index scheme for big spatio-temporal sensor data on cloud. In Proceedings of the 33rd IEEE International Geoscience and Remote Sensing Symposium, IGARSS, Melbourne, VIC, Australia, 21–26 July 2013; pp. 1713–1716.
14. Magdy, A.; Mokbel, M.F.; Elnikety, S.; Nath, S.; He, Y. Venus: Scalable real-time spatial queries on microblogs with adaptive load shedding. *IEEE Trans. Knowl. Data Eng.* **2016**, *28*, 356–370. [[CrossRef](#)]
15. Addair, T.G.; Dodge, D.A.; Walter, W.R.; Ruppert, S.D. Large-scale seismic signal analysis with Hadoop. *Comput. Geosci.* **2014**, *66*, 145–154. [[CrossRef](#)]
16. Zou, Z.Q.; Wang, Y.; Cao, K.; Qu, T.S.; Wang, Z.M. Semantic overlay network for large-scale spatial information indexing. *Comput. Geosci.* **2013**, *57*, 208–217. [[CrossRef](#)]
17. Jhummarwala, A.; Mazin, A.; Potdar, M.B. Geospatial Hadoop (GS-Hadoop) an efficient mapreduce based engine for distributed processing of shapefiles. In Proceedings of the the 2nd International Conference on Advances in Computing, Communication, & Automation, Bareilly, India, 30 September–1 October 2016; pp. 1–7.
18. Yao, X.; Li, G. Big spatial vector data management: A review. *Big Earth Data* **2018**, *2*, 108–129. [[CrossRef](#)]
19. OGC. Geographic Information-Well-Known Text Representation of Coordinate Reference Systems. Available online: <http://docs.opengeospatial.org/is/12-063r5/12-063r5.html> (accessed on 20 June 2018).
20. Zhao, L.; Chen, L.; Ranjan, R.; Choo, K.-K.R.; He, J. Geographical information system parallelization for spatial big data processing: A review. *Clust. Comput.* **2015**, *19*, 139–152. [[CrossRef](#)]
21. Singh, H.; Bawa, S. A mapreduce-based scalable discovery and indexing of structured big data. *Future Gener. Comput. Syst.* **2017**, *73*, 32–43. [[CrossRef](#)]
22. Yao, X.; Mokbel, M.F.; Alarabi, L.; Eldawy, A.; Yang, J.; Yun, W.; Li, L.; Ye, S.; Zhu, D. Spatial coding-based approach for partitioning big spatial data in Hadoop. *Comput. Geosci.* **2017**, *106*, 60–67. [[CrossRef](#)]
23. Hadjieleftheriou, M.; Manolopoulos, Y.; Theodoridis, Y.; Tsotras, V.J. R-trees—A dynamic index structure for spatial searching. In *Encyclopedia of GIS*; Shekhar, S., Xiong, H., Eds.; Springer: Boston, MA, USA, 2008; pp. 993–1002.
24. Eldawy, A.; Alarabi, L.; Mokbel, M.F. Spatial partitioning techniques in spatialhadoop. *Proc. VLDB Endow.* **2015**, *8*, 1602–1605. [[CrossRef](#)]
25. Zhang, J.; You, S. High-performance quadtree constructions on large-scale geospatial rasters using GPGPU parallel primitives. *Int. J. Geogr. Inf. Sci.* **2013**, *27*, 2207–2226. [[CrossRef](#)]
26. Eldawy, A.; Mokbel, M.F.; Jonathan, C. Hadoopviz: A mapreduce framework for extensible visualization of big spatial data. In Proceedings of the 32nd IEEE International Conference on Data Engineering, Helsinki, Finland, 16–20 May 2016; pp. 601–612.
27. Liu, Y.; Chen, L.; Jing, N.; Xiong, W. Parallel batch-building remote sensing images tile pyramid with mapreduce. *Wuhan Daxue Xuebao (Xinxi Kexue Ban)/Geomat. Inf. Sci. Wuhan Univ.* **2013**, *38*, 278–282.
28. Lin, W.; Zhou, H.; Xia, P. An effective NOSQL-based vector map tile management approach. *ISPRS Int. Geo-Inf.* **2016**, *5*, 1–25.
29. Lee, J.-G.; Kang, M. Geospatial big data: Challenges and opportunities. *Big Data Res.* **2015**, *2*, 74–81. [[CrossRef](#)]



© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).