Hindawi
Mathematical Problems in Engineering
Volume 2022, Article ID 8508273, 11 pages
https://doi.org/10.1155/2022/8508273

Research Article

Tunnel Security Management Based on Association Rule Mining under Hadoop Platform

Qun Wang[1] **and Ting Xue**[2]

[1]*Xi'an Siyuan University, Shaanxi, Xi'an 710038, China*
[2]*Xi'an Eurasia University, Shaanxi, Xi'an 710065, China*

Correspondence should be addressed to Ting Xue; xueting@eurasia.edu

Received 13 June 2022; Revised 7 July 2022; Accepted 16 July 2022; Published 10 August 2022

Academic Editor: Zaoli Yang

The problem of how to use large amounts of historical data for tunnel safety management has a greater practical application value. The association rule method in data mining technology can provide effective decision support for tunnel safety prevention by mining historical data. To address the problem of large data volume and sparse data items in tunnel safety management, an association rule method—Apriori algorithm—based on the Hadoop platform is proposed to improve the efficiency and accuracy of data mining in cloud environment. First, the parallel MapReduce implementation steps are analyzed on the basis of the distributed Hadoop framework. Then, the existing single-user data validation algorithm is improved by applying a multiuser parallel validation algorithm to Apriori in order to reduce the number of validations. Next, the traditional association rule Apriori algorithm is MapReduce optimized to generate a smaller set of useless candidate items. At the same time, Boolean ranking is used to optimize the way transactional data are stored in the database, reducing the number of redundant subsets and the number of times the database is connected, and shortening the task processing time. The experimental results show that the proposed method is able to mine the relationships between tunnel safety hazards and provide effective decision support for tunnel safety prevention. At the same time, the proposed method more efficiently operates than other association rule methods.

1. Introduction

China started building its first railway tunnels in 1888. According to statistics from 2004, more than 7,400 tunnels with a total length of 4,200 km were in operation in China's railways, but over time, a large number of tunnels have gradually started to present safety hazards. Statistics from the Chinese railway authorities in 2015 show that 67.5% of the tunnels in Sichuan province, for example, have varying degrees of safety hazards. Some of these safety hazards have endangered normal working safety and are tending to become more and more serious. The railway authorities test tunnel safety indicator data every year and have accumulated a large amount of historical data. However, these data are basically idle and do not play an effective role in the prediction of security problems.

At this stage, the traditional management system only performs some simple queries and statistical work on these collected data [1–5]. Although, every year, the railway authorities invest a lot of manpower and funds in the maintenance and protection of tunnels, the safety condition of tunnels still does not show any significant improvement. This is because such manual statistical methods cannot effectively uncover the safety risk patterns' latent in these data. Therefore, more advanced analytical techniques are needed to extract the safety risk characteristics of railway tunnels [6–8]. In addition, we need to find some subtle relationships hidden between a large amount of tunnel safety-related data, so as to provide strong decision support for the daily maintenance and safety inspection of railway tunnels [9, 10]. The association rule approach in data mining techniques is a powerful tool to help us achieve this.

Data mining is the process of extracting potentially useful information and knowledge from fuzzy, random real-world application data [11–14]. The data source must be real and noise-laden. Data mining discovers knowledge that is of interest to the user. In a broad understanding, concepts, rules, patterns, laws, and constraints are knowledge. Data can be structured, such as data in a relational database. Data can also be semistructured, such as textual, graphical, and image data. The knowledge found can be used for information management, query optimization, decision support, and process control. Thus, unlike low-level simple queries, data mining is a cross-cutting discipline [15–17] that can provide more advanced decision-making. In contrast to traditional analytical methods (e.g., query and online application analysis), data mining mines information and discovers knowledge without explicit assumptions.

The association rule mining is one of the key research directions in data mining and is used to discover interesting associations between sets of items from large amounts of data. A typical example of association rules is the shopping blue analysis. For example, in the transaction database of a supermarket, association rules exist as follows: 20% of customers buy both item A and item B in one shopping trip; 80% of customers who buy item A also buy item B. Association rules have a wide range of applications [18, 19]: in business, association rules can assist companies in market analysis and customer relationship management; in genetics in biology, association rules can be used to analyze the correlation between gene expressions; and in cyber security, association rules can reveal the relationship graph between different attacks. The classical Apriori algorithm, proposed in the early 1990s [20], offers a completely new way of working with data for analysis. The basic idea of Apriori is to find relationships between different items from a database of historical transactions. The core of Apriori is to discover relationships between items by repeatedly scanning the data [21]. "Google predicts the flu," "beer and nappies," and "predicting pregnant women" are typical examples of the association rule algorithm—Apriori. Djenouri et al. [22] effectively reduced the number of candidate itemsets by using the Apriori algorithm in the process of generating candidate itemsets, so the performance of data mining has been greatly improved. Li et al. [23] proposed a multilevel association rule mining algorithm—cumulate. The cumulate algorithm adds the ancestor of each item to the transaction database and then uses the Apriori algorithm to mine the extended database. The cumulate algorithm avoids the generation of redundant frequent patterns in the mining process. Javed et al. [24] proposed an improved algorithm based on Apriori. This algorithm starts mining at the higher levels of the concept hierarchy tree and then proceeds down to the lower levels in turn. The Apriori algorithm is still used when mining on different concept layers, but a different support threshold can be used for each concept layer. However, the above algorithm still needs to generate a large number of candidate sets and needs to repeatedly scan the transaction database to verify the frequency of these candidate sets, which seriously affects the efficiency of the algorithm.

There is no doubt that the weak storage and computational capacity of individual computer nodes can no longer meet the needs of big data mining. The Hadoop system has the two main advantages of distributed storage and parallel computing when dealing with large amounts of data. Developers can easily develop applications on the Hadoop platform [25–28], and the Hadoop system is able to perform distributed analysis of huge amounts of data on a cluster of computers using a simpler programming model. In recent years, there has been a proliferation of Hadoop-based applications. Yahoo built the largest Hadoop cluster at the time in February 2008 using 2,000 nodes, and Facebook built a Hadoop cluster to perform machine learning-based data analysis research. Alibaba (the world's largest B2C provider) built a Hadoop system to store and analyze huge amounts of transaction data. The Apriori algorithm in the association rule model is mainly used to find out frequent patterns or association relationships between items in a large amount of historical data. However, due to the inherent problems of the Apriori algorithm [29], when the volume of data is huge, it generates a complex computing process. At this point, continuing to use the Apriori algorithm to analyze and process the data would result in a less efficient system execution. Therefore, more and more researchers have started to try to improve the Apriori algorithm in order to improve the efficiency of data mining.

A similar problem exists in the historical tunnel safety data. Due to the complex topography of China and the fact that China is also a large tunnel country, the railway tunnel management has to carry out several inspections of railway tunnels across the country every year. The long-term inspection process has accumulated a large amount of historical tunnel safety data. However, these data have largely been left unused and have not played an effective role in the prediction of safety risks. The Hadoop system provides a new way to solve this problem.

The aim of this research is to implement data mining algorithms integrated with a Hadoop system and applied to the task of mining historical tunnel safety data. The proposed method can provide effective decision support for tunnel safety prevention by mining historical data.

The main innovations and contributions of this study include the following:

(1) Dividing the larger tunnel history security data database into several very small databases so that these smaller databases can be run under Hadoop. In other words, when the Apriori algorithm operates on large amounts of data, the large amount of tunnel history safety data is divided so that it can perform parallel computational operations.

(2) In order to run association rule mining on the Hadoop system framework, the existing single-user data validation algorithm was improved by applying the multiuser parallel validation algorithm to Apriori in order to reduce the number of validations.

(3) The traditional association rule Apriori algorithm is improved by reducing the number of redundant subsets and the number of times the database is

linked. Also, Boolean permutations are used to preprocess the optimized database, thus reducing the task processing time.

The rest of the study is organized as follows: In Section 2, the cloud-based Hadoop systems are studied in detail, while Section 3 provides the Hadoop-based optimization of association rule mining. Section 4 provides the experimental results and analysis. Finally, the study is concluded in Section 5.

2. Cloud-Based Hadoop Systems

2.1. Cloud Computing Theory and Related Technologies. Cloud computing is currently a research hotspot in computer science and technology, which has gained the attention of many enterprises and related internet experts, and is an important trend in the future development of computer network technology. The concept of cloud computing was first introduced by Google's CEO, Eric Schmidt, at the Internet Conference in 2006. A typical cloud computing platform needs to have (1) a gridded data storage matrix network; (2) firewall devices; and (3) computing resource devices. A cloud computing platform allows users to remotely use a scalable cloud storage space for cloud application services through a lease, as shown in Figure 1.

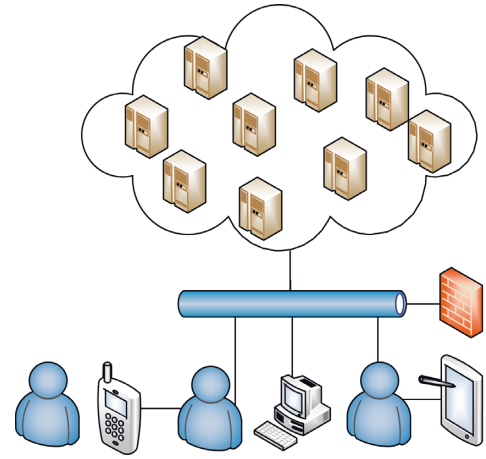

Figure 1: Principle of cloud computing services.

A complete cloud computing architecture should include the following: an access layer, a core layer, a resource aggregation layer, an API interface layer, and an application layer, as shown in Figure 2.

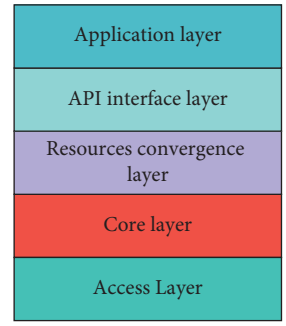

Figure 2: Cloud computing architecture.

2.2. Distributed Hadoop Framework. Hadoop is a framework consisting of different software libraries (also known as functional modules). The most important components of Hadoop include Common, HDFS, and MapReduce. The Hadoop cloud platform has shown excellent performance when solving big data mining problems. As one of the three major distributed computing systems, Hadoop can easily accomplish the fusion of different structural types of data. Hadoop can provide a computing environment with distributed storage across computer clusters. Hadoop has unique advantages in data analysis. Information resources in the big data environment are characterized by openness. In addition, considering the problem of big data throughput, the fluency of resource interaction is particularly important in the process of user behavior data mining. As big data is more frequently uploaded and downloaded, the Hadoop platform is better suited to the management of historical tunnel security data.

The core architecture of a Hadoop ecosystem is shown in Figure 3.

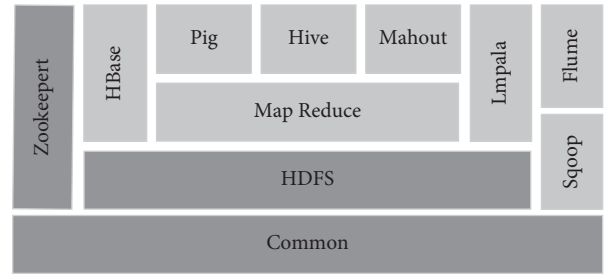

Figure 3: Distributed Hadoop framework.

Common is a functional module at the bottom of the Hadoop core architecture [30] and is the basis for other software libraries. Common provides tools for various Hadoop subprojects, such as system configuration tools, remote procedure calls, and serialization mechanisms. HBase is an open-source distributed database based on Hadoop Distributed File System (HDFS), and HBase has the advantage of being able to read and write large datasets randomly in real time. Hive is a data warehouse tool for Hadoop that maps structured data files into a single data table. Pig's data processing is transformed into a MapReduce job designed to analyze plain text input files. Lmpala is a unified platform for real-time queries. Flume is a distributed data collection system that addresses the collection, aggregation, and transfer of large volumes of logs.

2.3. Analysis of MapReduce Working Principles. First, we need to analyze the steps of a parallel MapReduce implementation based on the distributed Hadoop framework. MapReduce greatly simplifies and improves programming efficiency while ignoring the underlying hardware. For the tunnel history safety data database to work in a MapReduce model, it must be split. Each small piece of data after splitting

requires that it be able to perform operations independently and in parallel. Each MapReduce job performs a split of the input large dataset as it performs its operations. The MapReduce framework consists of a master job tracker and a number of Slave job trackers together, where the number of Slave job trackers depends on the number of nodes in the cluster. The MapReduce working in the mechanism of MapReduce is shown in Figure 4.

When data or information is exported, the Map function will apply a DAG map to save it to memory.

$$DAG = (W, E, DAG_{\text{info}})$$
$$W = \{W_{\text{name}}, \{\text{Map}\}, \{\text{Reduce}\}, \text{Param}, \text{Input}, \text{Output}\}, \quad (1)$$

where W denotes the output of the DAG graph, W_{name} denotes the name of the output sequence, (Map) denotes all Map tasks, (Reduce) denotes all Reduce tasks, Param denotes the task-related system configuration parameters, Input denotes the input sequence, and Output denotes the output sequence. e denotes the coefficient between the 2 jobs in MapReduce. DAG_{info} denotes labeled data.

$$E = (\text{Path}, \text{Start}TK, \text{End}TK), \quad (2)$$

where Path denotes the path where the memory buffer is saved, StartTk denotes the target of the current Map function, and EndTk is the target of the subsequent Map function.

3. Hadoop-Based Optimization of Association Rule Mining

3.1. Tunneling Security in Hadoop Mode. The cloud Hadoop platform breaks geographical and time constraints to unify the management of tunnel security. Hadoop stores tunnel security data on the cloud, thus enabling data sharing. The mining of data on Hadoop requires advanced association rule technology to support it, especially as the volume of tunnel safety data is large and the data items are sparse. The Hadoop-based tunnel security management model has the following three main changes: (1) the system architecture model has changed; (2) data sharing in the cloud computing environment has changed the traditional way of working with statistics; and (3) the security risks of tunnels have become more prominent. In order to improve the efficiency of association rule mining in the Hadoop model, a tunnel security data storage system in the cloud environment has been designed, as shown in Figure 5, to guarantee the information connectivity between data nodes.

3.2. Parallel Verification Algorithms. Traditional data validation algorithms can only complete data integrity validation for one user during a single validation calculation. The above analysis of the MapReduce working principle shows that this problem can seriously affect the efficiency of running association rule mining on the Hadoop framework. Therefore, a parallel validation algorithm is proposed in this study. The proposed parallel verification algorithm can

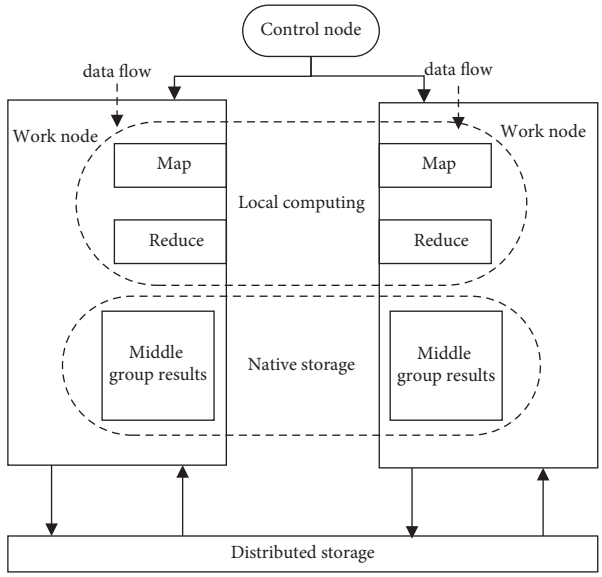

Figure 4: MapReduce working mechanism.

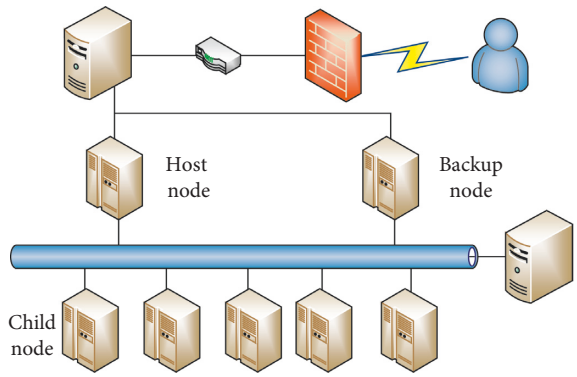

Figure 5: Cloud storage system network topology.

verify the data of multiple users in parallel during the computation of a single verification. The proposed parallel verification algorithm not only reduces the number of verifications in the verification computation but also reduces the data transfer bandwidth required for authentication between the user (third-party verifier) and the cloud storage. The cloud storage security architecture is shown in Figure 6.

The specific steps of the proposed parallel validation algorithm are as follows:

Step 1. Generate relevant files. Matching secret keys (public and private) are generated for all users and stored in a third-party trusted certification authority. We set the file signature for each file block in the verification file $F = (m_1, m_2, \ldots, m_n)$ to σ_i.

$$\sigma_i = \left(H(i) \cdot u^{m_i}\right)^x, \quad (3)$$

where $H(i)$ is a hash function, and x and u are both random numbers.

Step 2. Send a validation request. We generate a sequence $CHAL$ of validation requests for each file block.

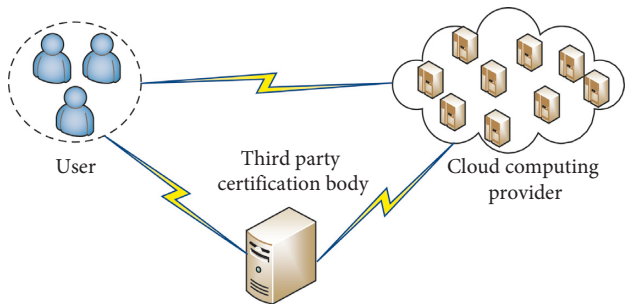

Figure 6: Cloud computing storage security architecture.

$$CHAL = \{(i, v_i)\}, \tag{4}$$

where i is the sequence number of the file block, and v_i is the corresponding random number.

Step 3. Generate authentication messages. In order to achieve multiuser parallel authentication, it is first necessary to complete the grouping and merging of all user files first.

$$\mu_k = \sum_{i=1}^{n} v_i m_{k,i} + \mu_r, \tag{5}$$

where μ_r represents the random number generated for each user during each verification by the cloud storage server. The signature is then computed.

$$\sigma = \prod_{k=1}^{K} \left(\prod_{i=1}^{n} \sigma_{k,i}^{v_i} \cdot r_k \right), \tag{6}$$

where $\sigma_{k,i}^{v_i}$ indicates the digital signature of a single document, and $r_k = \mu_k^{\mu_r}$.

Step 4. Complete verification of the results of the group merge and determine whether the cloud storage is correct.

3.3. Database Preprocessing Optimization. In this study, the rotation method is used to realize the partition operation of a parallel database. We scan the relationship sequentially and store the ith tuple on the disk labeled Dimod, thus ensuring that the tuples are evenly distributed on multiple disks. The Hadoop cloud platform has shown excellent performance in solving such big data mining problems, but centralized processing can easily cause network congestion problems as data become more complex and databases become larger. As a result, traditional cloud computing systems are no longer able to effectively solve big data processing tasks. Currently, parallel MapReduce job stream processing techniques on a distributed Hadoop platform are the mainstay of research today. In order to effectively implement Apriori on the distributed Hadoop platform and further improve the accuracy of recommendations, an Apriori optimization algorithm based on the Hadoop platform is proposed.

A variety of improvements to the Apriori algorithm have been proposed by many. Most of the improvement algorithms increase the speed but decrease the accuracy. For example, data filtering improvement algorithms often filter out some important data, which affects the final frequent set obtained. Other improvement algorithms, such as the hash tree technique, focus only on running the algorithm process. Huge hash trees can be problematic when dealing with collisions. Almost all of the improved algorithms focus on the algorithm itself and often neglect to look at database preprocessing.

In the first step of the traditional Apriori algorithm, multiple scans of the database are required. When the amount of data stored in the database is large, this way of recording and mapping the data takes a lot of time. In order to reduce the number of database scans and thus improve the efficiency of the algorithm, this study uses Boolean alignment to preprocess and optimize the way in which transactional data are stored in the database. When reading transactional information from the database, the method only needs to perform one scan of the database, which greatly improves the efficiency of the algorithm. Let I be the set of items and $\mathbf{D}$ be the set of task-related transactions in the database. Each transaction T is the set of items satisfying the condition $T \subseteq I$. The matrix of the set of items I is shown as follows:

$$\mathbf{D} = (\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_n) = \begin{bmatrix} d_{11} & d_{12} & \cdots & d_{1n} \\ d_{21} & d_{22} & \cdots & d_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ d_{m1} & d_{m2} & \cdots & d_{mn} \end{bmatrix}, \tag{7}$$

where m is the total number of transactions, and n is the total number of items.

The vector of each term I_j is calculated as follows:

$$\mathbf{D}_j = \begin{bmatrix} \alpha_{1j} \\ \alpha_{2j} \\ \vdots \\ \alpha_{pj} \end{bmatrix}, \tag{8}$$

where α_{ij} is defined as follows:

$$\alpha_{ij} = \begin{cases} 0 I_j \notin T_i \\ 1 I_j \in T_i \end{cases}. \tag{9}$$

Therefore, the support of I_j for support (I_j) is calculated as follows:

$$\text{support}\left(I_j\right) = \sum_{i=1}^{p} \alpha_{ij}. \tag{10}$$

In addition, we can obtain the k-item set support, which is also a fuzzy criterion for multiattribute decision-making.

$$\text{support}\ (I_i, I_2, \ldots, I_k) = \sum_{t=1}^{p} \{(d_{t1} \wedge d_{t2} \wedge \ldots \wedge d_{tk-1}) d_{tk}\}, \tag{11}$$

where the symbol "∧" indicates a logic operation.

3.4. MapReduce Parallelization of the Apriori Algorithm. The traditional Apriori algorithm steps require repeated execution of the join and scan database operations, which

significantly reduces the efficiency of the execution. Therefore, in order to reduce the number of redundant subsets and databases being concatenated, in this study, the process (join step) for generating the candidate itemset$C_k$has been improved as shown in Table 1.

In order to implement the optimized Apriori algorithm concretely in the Hadoop framework, the traditional Apriori algorithm process is optimized for MapReduce processing in this study, as shown in Figure 7.

With the above steps, it is possible to reduce the number of database connections (only one scan is required) while generating as few useless candidate sets as possible in order to improve the accuracy and efficiency of the association rule mining. Although the time complexity of the new algorithm has increased, the computational efficiency has been greatly improved. The new algorithm is significantly more efficient in memory operations than the traditional algorithm for database I/O operations.

The execution process of the MapReduce function mainly includes two parts:

(1) The execution of the Map function. A Slave assigned with map task reads a data block to get key/value pairs, and then, the Map function processes these < TID, ITEM > to get local L_k.

(2) The execution of Reduce tasks. When all the data blocks are analyzed, Reduce Slave starts to work. According to the arrangement of Master, each Reduce reads the files (local frequent itemsets) in the local disk of Map Slave, and generates the global frequent itemsets after calculating and merging by the Reduce function.

4. Experimental Results and Analysis

4.1. Experimental Environment Setup. There are three ways to build Hadoop [31]: local mode, pseudodistributed mode (the Java processes corresponding to Hadoop run on a single physical machine), and clustered mode (the Java processes corresponding to Hadoop run on multiple physical machines). Since the cluster mode built on a virtual machine is close to the actual production environment, we will use this deployment mode to build the Hadoop system in the following. This experiment uses 20 servers to build the Hadoop runtime environment, and all 20 servers use CentOS 7 as the operating system. We take six servers as an example, and the hardware configuration and IP and other information are shown in Tables 2 and 3.

We use the WinSCP tool to upload the jdk-8u45-linux-i586.bin downloaded from Oracle's official website to the/usr/local directory of the Linux operating system. We unzip and complete the JDK8 installation, then configure the following environment variables in the profile file in the etc directory: export JAVA_HOME = /home/jdk; export PATH = $JAVA_HOME/bin:$PATH. save and exit, and we use the source/etc/profile command to refresh the environment variables you just configured. To ensure that the hostname and IP address of each node in the cluster are correctly resolved, we need to bind each node hostname to its corresponding IP address. Execute the #ssh-keygen-t command on each machine to generate an encrypted file. Execute the scp /root/.ssh/authorized_keys command on the namenode to generate the configuration file.

Table 1: Optimization algorithm for association rules.

```
(1) for each lx Lk- 1;
(2) {
(3)       for each ly lk- 1;
(4)       {
(5)       if (lx [1] = ly [1]∧lx [2] = ly [2]∧lx [k- 2] = ly [k-2]∧lx [k- 1]
<ly [ k- 1])
(6)          {
(7)             c = lx ∪ ly; //Complete the concatenation of the two
item sets;
(8)             ck = c ∪ ck;
(9)          }
(10)       else break;
(11)    }
(12) }
(13) end for
(14) return ck//Returns the item sets of generated by the join;
```

The first time you start Hadoop, you must first format the NameNode node. On the master node Hadoop0, we execute the #hadoop namenode format command to manipulate the formatting. Then, we use the command to start the cluster, as shown in Figure 8.

4.2. Results of Mining Railway Tunnel Safety Data. The experimental data were derived from tunnel safety inspection data provided by the Sichuan Railway Bureau in China. A total of 2,787 tunnel safety inspection data in total were available in 2016. The raw data cannot be directly analyzed by association rule mining. Prior to the data mining work, preprocessing of the data, such as removing redundant information from the data based on intuitive experience, is required. The final training set obtained for the 2016 tunnel safety data mining part is shown in Table 4.

The Hadoop-based association rule algorithm was used to mine the relationships between security hazards at a support level of 0.3%. After processing by Java program, frequent l-item sets were obtained by scanning the data. The training set was also sorted according to the value of the l-item set from largest to smallest, while pruning (removing the nonfrequent l-item set from the training set) was performed. The key safety hazards of the tunnels that were mined are shown in Table 5.

As can be seen from the excavation results shown in Table 6, most railway tunnels occur with more than one safety hazard. These safety hazards occur in groups. One safety hazard can often cause the occurrence of another or several safety hazards, as shown in Figure 9. The traditional approach is to treat the safety hazards according to their severity, which makes them ineffective and inefficient. If the railways were able to treat the hazards sequentially and in parallel according to their association, this would greatly improve the effectiveness and efficiency of the treatment.

Compared to traditional simple statistical and query work, we have obtained correlation results that are not

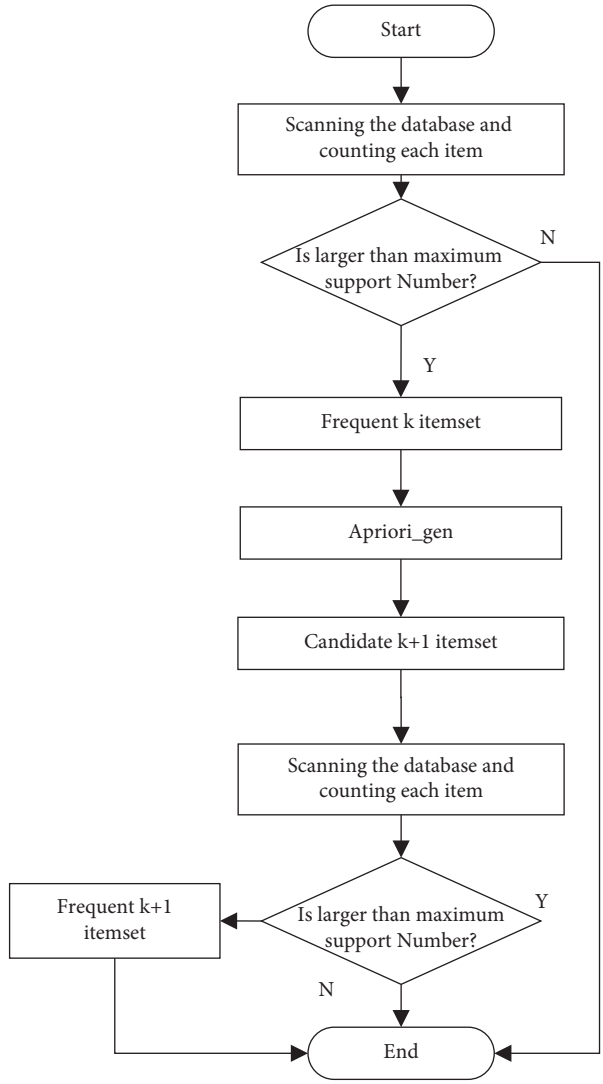

Figure 7: Apriori algorithm optimization based on MapReduce.

available with data comparison or query statistics through improved association rule mining algorithms, and found some hidden knowledge in the data, which will provide effective decision support for railway tunnel management departments.

4.3. Validation of Database Preprocessing. The performance comparison results for the data parallel verification algorithm are shown in Figure 10.

It can be seen that as the number of nodes in the Hadoop environment increases, the average time required to update each file remains constant in the traditional single-user verification algorithm, but the average time required to update each file gradually decreases in the multiuser parallel verification algorithm. Therefore, the parallel verification algorithm is computationally more efficient than the single-user verification algorithm in a shared Hadoop-based model.

4.4. Data Comparison Tests. A cluster of 20 nodes was used in this experiment to test the performance of the algorithm before and after improvements. Transactions in the dataset are stored as files. Each transaction occupies one record in the file. The data items contained in each transaction are separated from each other by tabs. The traditional serial Apriori algorithm and the MapReduceized improved algorithm of this study are set to the same support of 0.3% when processing the dataset. The results of several experiments with a constant number of nodes and increasing number of transactions are shown in Table 6.

Based on the results of the experiments in the table above, a line graph of the traditional algorithm and the MapReduce parallelization algorithm is derived, as shown in Figure 11.

At support levels of 0.4%, 0.5%, 0.6%, 0.8%, 1.0%, 1.2%, and 1.4%, respectively, the traditional serial Apriori algorithm and the improved Apriori algorithm found the same frequency patterns, and therefore, the improved Apriori algorithm was feasible. A comparison of the running times of the traditional serial Apriori algorithm, the MFP-growth algorithm [32], and the improved Apriori algorithm with a constant database and varying support is shown in Figure 12.

Table 2: Machine hardware configuration information.

	CPU	Memory (G)	Hard disk	NICs
Hadoop0	Intel Xeon E5-2620	32	4 TB SATA +256 GB	Intel 82574L
Hadoop1	Intel Xeon E5-2620	32	2TBSATA +128 GB	Intel 82574L
Hadoop2	Intel Xeon E5-2620	32	1 TB SATA+128 GB	Intel 82574L
Hadoop3	Intel Xeon E5-2620	32	1 TB SATA+128 GB	Intel 82574L
Hadoop4	Intel Xeon E5-2620	32	1 TB SATA+128 GB	Intel 82574L
Hadoop5	Intel Xeon E5-2620	32	1 TB SATA+128 GB	Intel 82574L

Table 3: Node allocation information table.

	Allocation information	IP
Hadoop0	Master	10.10.108.50
Hadoop1	Slave	10.10.108.62
Hadoop2	Slave	10.10.108.63
Hadoop3	Slave	10.10.108.64
Hadoop4	Slave	12.12.12.74
Hadoop5	Slave	12.12.12.75

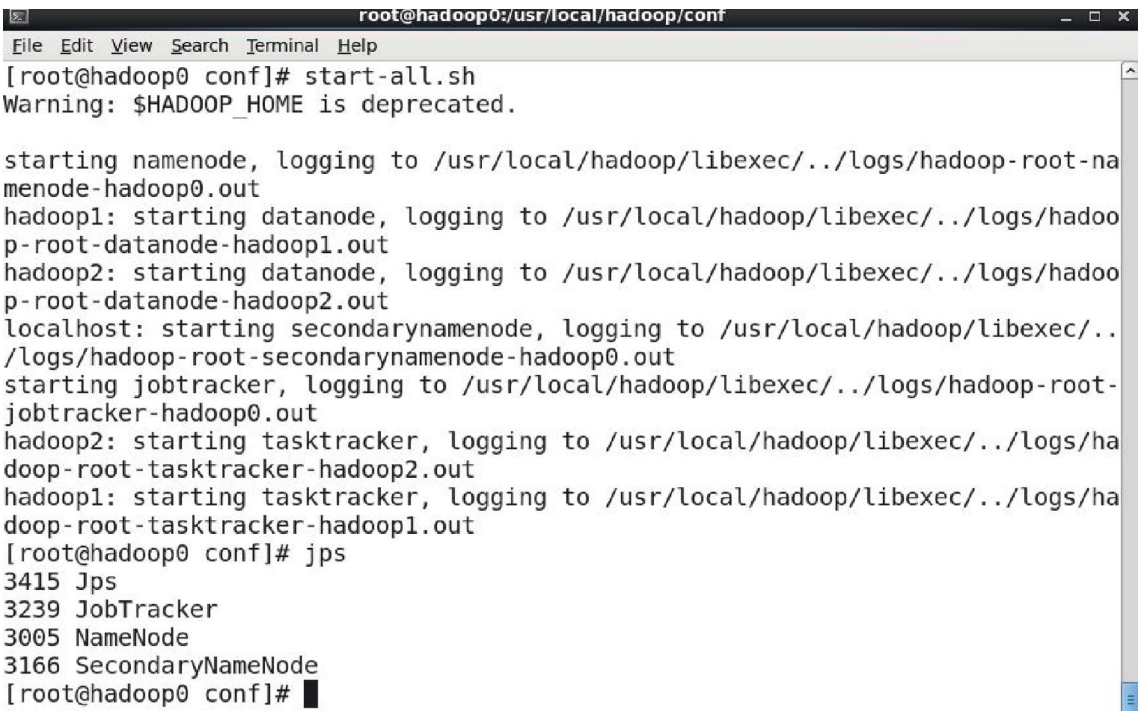

Figure 8: Starting a Hadoop cluster from the command line.

Table 4: Selected training sets for tunnel safety data mining in 2016.

Tunnel number	Safety hazard code	Association rule codes	TID
1	"S0102, S0202, S0303, S0401, S0501"	2 5 8 9 12	T001
2	"s0102, s0103, s0202, s0302, s0401, s0403"	2 3 5 7 9 11	T002
3	"s0101, s0103, s0303, s0304, s0403"	1 3 8 10 11	T003
4	"s0101, s0202, s0302, s0401, s0501"	1 5 7 9 12	T004
5	"S0102, S0202, S0303, S0401, S0501"	2 5 8 9 12	T005
...	...	...	...

Table 5: Key safety hazards in tunnels.

Association rule codes	Safety hazard code	Name of safety hazard
2	S0102	Cracking in the wall
5	S0202	Water leaks
9	S0401	Gauge limits
12	S0501	Damage to the base of the paving
8	S0303	Slumped rock fall at the cave entrance

Table 6: Performance comparison before and after algorithm improvement.

Number of transaction records	Traditional serial Apriori	Improved Apriori
2000	6.351	6.406
4000	9.167	9.002
6000	12.592	11.166
8000	15.036	13.421
10000	17.809	15.914
12000	21.134	19.037
14000	23.028	20.135
16000	26.167	23.443
18000	29.472	25.128
20,000	35.503	31.601

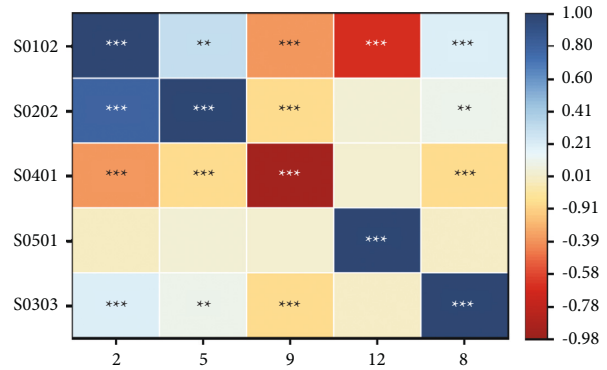

Figure 9: Statistical analysis between association rule codes and safety hazard codes.

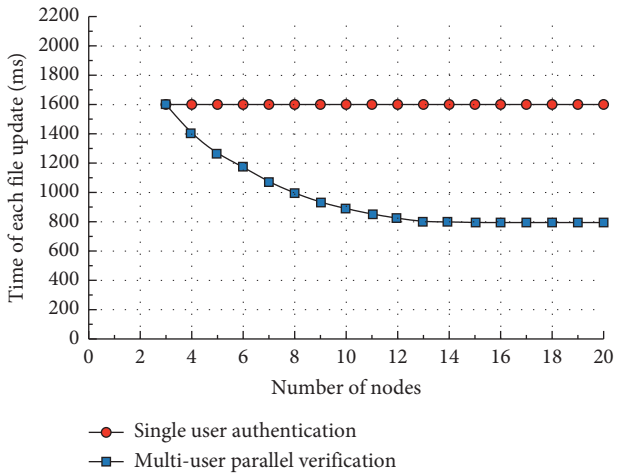

Figure 10: Performance comparison of data integrity verification algorithms.

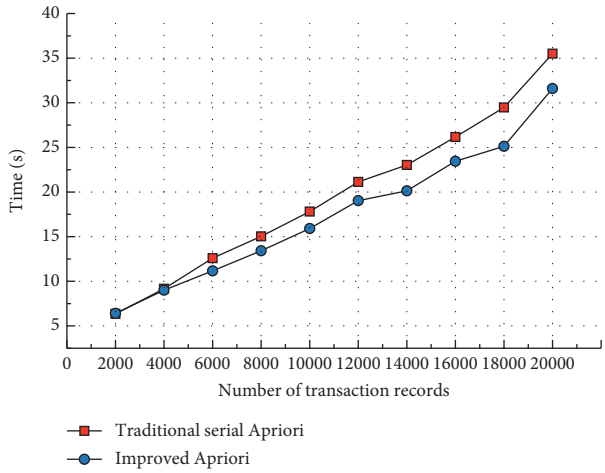

Figure 11: Comparison of traditional algorithms with MapReduce parallelization algorithms.

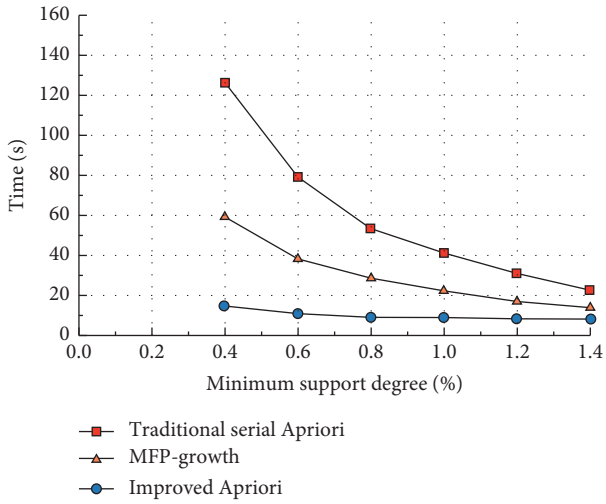

Figure 12: Running time of each algorithm.

It can be seen that the smaller the support degree, the more obvious the performance advantage of the proposed improved Apriori algorithm.

5. Conclusions

To address the tunnel safety management problem, this study proposes a MapReduce parallelized Apriori algorithm based on the Hadoop platform. By integrating association rule data mining techniques with the Hadoop system, the relationships between tunnel safety hazards are mined. The larger database of historical tunnel safety data is divided into several very small databases so that these smaller databases can run under Hadoop. In order to run association rule mining on the Hadoop system framework, the existing single-user data validation algorithm was improved by applying a multiuser parallel validation algorithm to Apriori in order to reduce the number of validations. The following conclusions were drawn.

(1) There is more than one safety hazard that occurs in most railway tunnels. One safety hazard can often lead to the occurrence of another or several safety hazards. If the railway authorities can treat the defects sequentially and in parallel according to their association, this will greatly improve the effectiveness and efficiency of the treatment.

(2) Parallel verification algorithms have higher computational efficiency compared to single-user verification algorithms in a shared Hadoop-based model.

(3) Compared with other association rule methods, the MapReduce parallelized Apriori algorithm runs more efficiently. The smaller the support, the more obvious the performance advantage of the proposed method.

Data Availability

The experimental data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] S. He, B. Liang, G. Pan, F. Wang, and L. Cui, "Influence of dynamic highway tunnel lighting environment on driving safety based on eye movement parameters of the driver," *Tunnelling and Underground Space Technology*, vol. 67, pp. 52–60, 2017.

[2] P. Bernardi, E. Michelini, A. Sirico, S. Rainieri, and C. Corradi, "Simulation methodology for the assessment of the structural safety of concrete tunnel linings based on CFD fire - FE thermo-mechanical analysis: a case study," *Engineering Structures*, vol. 225, Article ID 111193, 2020.

[3] L. W. Zhang, H. Fu, J. Wu, X. Y. Zhang, and D. K. Zhao, "Effects of karst cave shape on the stability and minimum safety thickness of tunnel surrounding rock," *International Journal of Geomechanics*, vol. 21, no. 9, Article ID 04021150, 2021.

[4] G. H. Zhang, Y. Y. Jiao, L. B. Chen, H. Wang, and S. C. Li, "Analytical model for assessing collapse risk during mountain tunnel construction," *Canadian Geotechnical Journal*, vol. 53, no. 2, pp. 326–342, 2016.

[5] R. Klein, I. Maevski, J. Ko, and Y. Li, "Fuel pool development in tunnel and drainage as a means to mitigate tunnel fire size," *Fire Safety Journal*, vol. 97, no. 4, pp. 87–95, 2018.

[6] Y. Tao, M. Yeung, and Y. Bing, "Three-dimensional stability of landslides based on local safety factor," *Journal of Mountain Science*, vol. 13, no. 9, pp. 1515–1526, 2016.

[7] H. Sun, S. Song, W. Lu, Q. Ren, and X. Li, "Safety design of tunnel lining structure considering bond-slip failure mechanism," *Arabian Journal of Geosciences*, vol. 15, no. 6, pp. 1–12, 2022.

[8] H. Ingason, "New challenges in tunnel fire safety," *Fire Technology*, vol. 52, no. 5, pp. 1445–1447, 2016.

[9] M. Turbatu, L. Stroica, A. M. Oproiu, A. Barbilian, C. Feng-Ifrim, and A. T. Ispas, "Carpal tunnel syndrome: surgical landmarks which increase safety in carpal tunnel release procedures," *Revista de Chimie*, vol. 71, no. 1, pp. 426–429, 2020.

[10] X. Li, X. Li, and Y. Su, "A hybrid approach combining uniform design and support vector machine to probabilistic tunnel stability assessment," *Structural Safety*, vol. 61, pp. 22–42, 2016.

[11] H. Hong, P. Tsangaratos, I. Ilia, J. Liu, A. X. Zhu, and W. Chen, "Application of fuzzy weight of evidence and data mining techniques in construction of flood susceptibility map of Poyang County, China," *Science of the Total Environment*, vol. 625, no. 6, pp. 575–588, 2018.

[12] R. M. Bommisetty, A. Khare, M. Khare, M. Khare, and P. Palanisamy, "Content-based video retrieval using integration of curvelet transform and simple linear iterative clustering," *International Journal of Image and Graphics*, vol. 132, no. 16, pp. 6–9, 2021.

[13] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.

[14] J. Lee, N. Ohba, and R. Asahi, "Discovery of zirconium dioxides for the design of better oxygen-ion conductors using efficient algorithms beyond data mining," *RSC Advances*, vol. 8, no. 45, Article ID 25534, 2018.

[15] D. Martins, G. M. Vianna, I. Rosseti, S. L. Martins, and A. Plastino, "Making a state-of-the-art heuristic faster with data mining," *Annals of Operations Research*, vol. 263, no. 1-2, pp. 141–162, 2018.

[16] A. Mohamed, J. Molendijk, and M. M. Hill, "Lipidr: a software tool for data mining and analysis of lipidomics datasets," *Journal of Proteome Research*, vol. 19, no. 7, pp. 2890–2897, 2020.

[17] Y. Djenouri, A. Belhadi, and R. Belkebir, "Bees swarm optimization guided by data mining techniques for document information retrieval," *Expert Systems with Applications*, vol. 94, no. 3, pp. 126–136, 2018.

[18] Y. Wang and L. Feng, "Improved adaboost algorithm for classification based on noise confidence degree and weighted feature selection," *IEEE Access*, vol. 8, Article ID 153011, 2020.

[19] A. Agrawal and A. Choudhary, "An online tool for predicting fatigue strength of steel alloys based on ensemble data mining," *International Journal of Fatigue*, vol. 113, no. 8, pp. 389–400, 2018.

[20] Y. Guo, M. Wang, and X. Li, "Application of an improved Apriori algorithm in a mobile e-commerce recommendation system," *Industrial Management & Data Systems*, vol. 117, no. 2, pp. 287–303, 2017.

[21] P. A. Jorge, D. Marcio, and I. P. Mario, "A multi-objective gene clustering algorithm guided by apriori biological knowledge with intensification and diversification strategies," *BioData Mining*, vol. 11, no. 1, pp. 16–22, 2018.

[22] Y. Djenouri and M. Comuzzi, "Combining Apriori heuristic and bio-inspired algorithms for solving the frequent itemsets mining problem," *Information Sciences*, vol. 420, pp. 1–15, 2017.

[23] L. Li, R. Lu, K. K. R. Choo, A. Datta, and J. Shao, "Privacy-preserving-outsourced association rule mining on vertically partitioned databases," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 8, pp. 1847–1861, 2016.

[24] M. F. Javed, W. Nawaz, and K. U. Khan, "HOVA-FPPM: flexible periodic pattern mining in time series databases using hashed occurrence vectors and apriori approach," *Scientific Programming*, vol. 2021, no. 1, 14 pages, Article ID 8841188, 2021.

[25] X. Xu, L. Cao, and X. Wang, "Adaptive task scheduling strategy based on dynamic workload adjustment for heterogeneous hadoop clusters," *IEEE Systems Journal*, vol. 10, no. 2, pp. 471–482, 2016.

[26] Z. Li, C. Yang, K. Liu, F. Hu, and B. Jin, "Automatic scaling hadoop in the cloud for efficient process of big geospatial data," *ISPRS International Journal of Geo-Information*, vol. 5, no. 10, p. 173, 2016.

[27] V. J. Hodge, S. O'Keefe, and J. Austin, "Hadoop neural network for parallel and distributed feature selection," *Neural Networks*, vol. 78, pp. 24–35, 2016.

[28] E. Dede, B. Sendir, P. Kuzlu, J. Weachock, M. Govindaraju, and L. Ramakrishnan, "Processing cassandra datasets with hadoop-streaming based approaches," *IEEE Transactions on Services Computing*, vol. 9, no. 1, pp. 46–58, 2016.

[29] A. K. Joshi, M. K. Brahmi, and B. K. Gupta, "Apriori to agricultural problems emerged through participatory rural appraisal in temperate zone of the Himalayas," *Indian Journal of Agricultural Sciences*, vol. 89, no. 2, pp. 220–230, 2019.

[30] H. Paul, C. Amandeep, and C. Andrew, "cl-dash: rapid configuration and deployment of Hadoop clusters for bioinformatics research in the cloud," *Bioinformatics*, vol. 32, no. 2, pp. 301–303, 2016.

[31] X. Cai, F. Li, P. Li, L. Ju, and Z. Jia, "SLA-aware energy-efficient scheduling scheme for Hadoop YARN," *The Journal of Supercomputing*, vol. 73, no. 8, pp. 3526–3546, 2016.

[32] S. Mai, M. Badawi, and S. El-Ghamrawy, "An optimized FP-growth algorithm for discovery of association rules," *The Journal of Supercomputing*, vol. 78, no. 4, pp. 5479–5506, 2021.