ISBN: 978-981-18-3959-7
2022 the 12th International Workshop on Computer Science and Engineering (WCSE 2022)
doi: 10.18178/wcse.2022.06.015

# Application of a Log Optimization Algorithm in the Earthquake Precursor System Data Synchronization

Wang Chen [1], Teng Yuntian [1] and Liu Qingjie [2+]

[1] Institute of Geophysics, China Earthquake Administration, Beijing, China

[2] School of Information Engineering, Institute of Disaster Prevention, Sanhe, Hebei, China

**Abstract:** In the earthquake precursor network observation system based on user-defined operation log for database synchronization, there were multiple preprocessed and re-collection of data in the station and regional center database, and there were multiple update and deletion records of the same preprocessing data and collected data in the operation log table. The logging of multiple operations for the same data will affect the efficiency of precursor data exchange. Aiming at this problem, we in the latest earthquake precursory network data exchange scheme, proposed an optimization algorithm of a custom action log, merge the custom operation log, by calculation in the operating log table, only keep the data synchronization needs the data of the last operation records, thus greatly reduce the network data flow, improve the efficiency of database synchronization.

**Keywords:** data exchange, operation log, log optimization, data transaction isolation.

## 1. Introduction

In the current earthquake precursor network observation system, data synchronization between databases depends on the original database synchronization mechanism of Oracle database which is realized by Oracle DB-link and trigger. Because the path of data exchange is complex, the data source is many, the data source type is complex, coupled with the need to cooperate with the registration and cancellation mechanism of the station, it is easy to cause the trigger deadlock phenomenon caused by inconsistent data, resulting in the channel blocking of DB-link, so that the data exchange fails[1]. In the latest precursor data management system based on the N-layer architecture design, we use a new data exchange strategy, which is based on user’s database operation logs and can realize the whole process of self-control data exchange strategy. This method of data synchronization based on database access logs can more accurately and efficiently replace the traditional data synchronization method, and has been adopted by many database business systems. Data synchronization, in essence, only data update operation, the traditional data synchronization scheme in accordance with the time sequence is the main library will change all of the data is passed to the synchronous library, due to the business characteristics of precursory data, the same observation data are are often preprocessed and updated many times , means that is updated every time the data synchronization, caused great processing overhead. By integrating and compressing the update logs of the database, the new algorithm keeps only the information related to data synchronization in the update logs, thus greatly reducing the data flow, reducing the processing overhead and improving the exchange efficiency, which has a certain popularization value.

## 2. Data Exchange and Data Operation Log

### 2.1. The Path of Data Synchronization

In the distributed seismic precursor network observation system, synchronization between databases is a core technology of the observation system. In the earthquake precursor network observation system, the data is synchronized from the non-node station database to the node station database, from the node station

[+] Corresponding author. Tel.: +86 13700349001;
*E-mail address*:21785576@qq.com.

database to the regional center database, from the regional center database to the national center database, and from the national center database to the national center database. Each subject center database, the subject center forms product data, and returns to the national center database[2]. The figure below shows the partial data flow path of the precursory network observation system data synchronization.

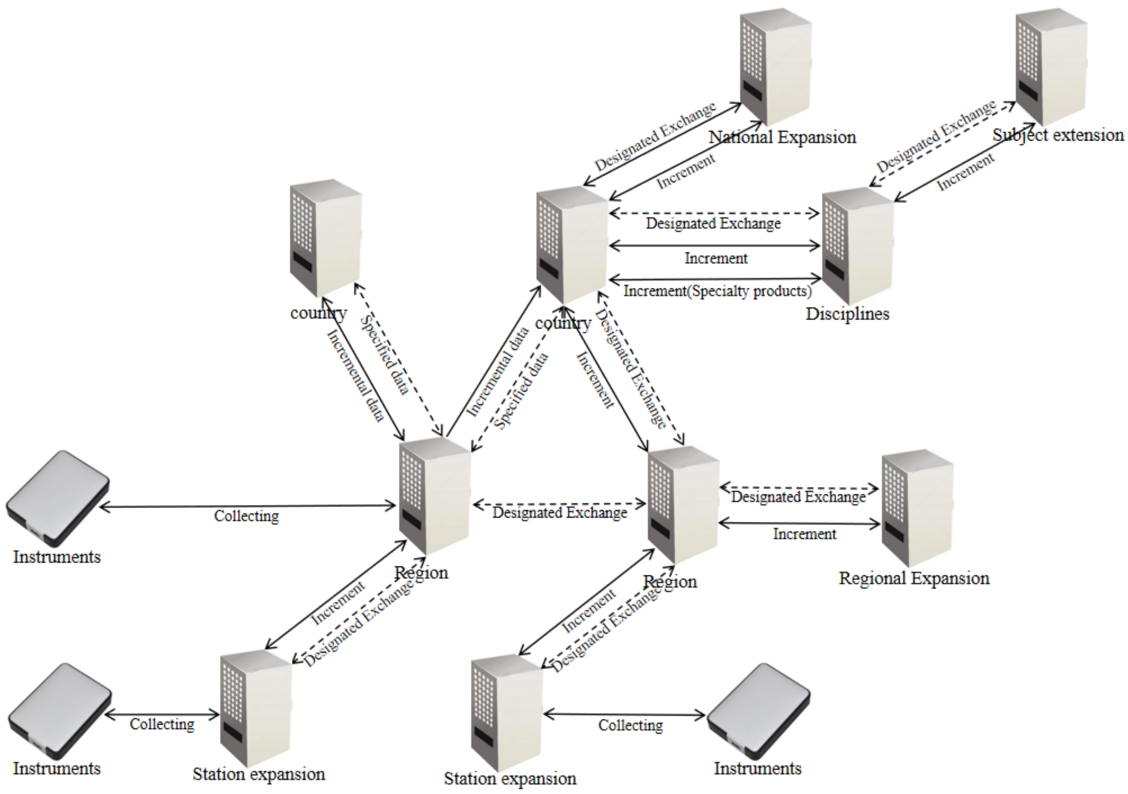


Fig.1: Data synchronization path diagram of precursory observation system

As can be seen from the above figure, the earthquake precursor network observation system belongs to a typical two-way exchange data source business system.

## 2.2. Time Synchronization of Data Exchange

Database exchange is divided into real-time exchange and asynchronous exchange according to the exchange time. Synchronous data synchronization is a technology that synchronizes the changed data to the target database in real time once the data source is updated. Synchronous data synchronization can ensure real-time data consistency between the data source and the data destination. In a distributed environment, it can ensure the consistency of the application and reduce the complexity of the application. When data changes occur at the data source, the changed data is captured immediately and propagated to the data destination. Therefore, real-time data synchronization technology is generally applied to data synchronization in a local area network or private network environment with relatively low data update frequency, relatively small data change and high bandwidth[3].

Asynchronous exchange is a database exchange technology that synchronizes data update-related operations such as data saving, deletion, and modification in a delayed manner. It has high reliability and flexibility of on-demand synchronization, but it is more complicated to implement than real-time data synchronization technology. Using asynchronous data synchronization, the log of incremental data is stored in the synchronization source database, and users can flexibly synchronize all or part of the data according to various conditions such as network conditions, data priorities, paths, and user needs[4].

The earthquake precursor network observation system is a large-scale distributed application system across the country with low bandwidth, high data output, multi-path, and frequent synchronization content changes[5]. Therefore, the use of asynchronous data synchronization technology is more in line with the needs and practical requirements of earthquake precursor observation. network conditions.

## 2.3. Application of Log Optimization Technology in Database Products

Data synchronization between distributed databases through database operation log technology is widely used in major mainstream database products: Sybase's copy customization, DB2's data connection bridge, Oracle's refresh through DB-Link snapshots, etc[6]. These technologies acquire incremental data by capturing incremental logs, and then perform one-way or two-way data synchronization at both ends or multiple ends of synchronization. Log-based database synchronization is widely used by major mainstream databases, which has become the actual standard of database synchronization and the key to flexible, reliable, effective and extended data synchronization.

The efficiency of data synchronization has a critical impact on the availability of various application systems. Conventional database logs provided by database manufacturers store the various versions of data involved in each piece of changed data. When these database logs are propagated to synchronization After the destination database, the single-record data will be frequently and invalidly modified, and only the last modification will be valid, resulting in a large amount of invalid data transmission and waste of processing resources[7]. Since only the last log is valid for data modification, if the database log can be optimized, the efficiency of data synchronization can be improved.

Earthquake precursory network observation system database synchronization needs path, synchronization, more frequent adjustment and network conditions, the content of the source and target land registration cancellation is frequent, unidirectional and bidirectional synchronous coexist, point-to-point synchronization and point to multi-point synchronous, partial synchronization with all different, yet database vendors provide the log operation interface has certain black box, You cannot perform precise log operations to meet application requirements. Therefore, in practical applications, customized database logs are used to capture data changes, and customized log operation interfaces and data operation interfaces are used to achieve a data synchronization solution with high flexibility, scalability, and reliability. Such user-defined database operation logs are also called virtual logs or mock logs[8-9].

# 3. Principles and Steps Of Log Optimization

## 3.1. Database Log

User-defined operation log is the basis of precursor data exchange, which is a collection of records of incremental data update activities in the entire database. For any update operation (such as insert, update, delete) of any data that needs to be exchanged, the primary key, table name, action, station number, measuring point number and other information of the data corresponding to the database operation must be saved and recorded in Virtual log action table[10].

The interceptor function implemented by the access component of the unified data access platform subsystem of the earthquake precursor network can import the data read and write logs into the virtual log operation table by checking the synchronization flag of the data access interface[11-12]. Define the virtual log table as P, and the unique identifier of the data as id, then the update sequence of the data table can be expressed as Sequence (T, v), and by compressing Sequence (T, v), it can effectively reduce the data synchronization process. amount of data transfer.

Define the keyword in table T as v The record of tv is tv, and the modification to tv can only be insert, update, delete, and the state transition of all possible modifications to tv is shown in Fig.2.

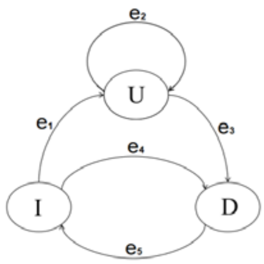


Fig.2: Operation log state transition diagram

Among them, the node represents the possible modification operations (insert, update, delete) to the tv; the vector edge e represents the possible modification state transition to the tv. Vector edge e4 means tv data is deleted; vector edge e2 means tv May be updated several times. Define a set M {m}, which represents the set composed of record state transitions in the above figure.

M{m}={ {I—>D},
{I —> U…U},
{I —> U…U —> D},
{D —> I —> U…U},
{U…U},
{U…U —>D},
{U…U —> D—>I}
}

Among them, represents the N data operations performed on the record tv; m is a set of possible operations on the record tv. Then, the Sequence (T, v) of any state can be hashed as a set of N m.

## 3.2. Optimization Principle

Assign values to the three operation nodes of insert, delete and update, define I=1, U=0, D=-1, define Value(m) as the sum of all node values in the path. Then, there are the following conclusions:

(1) If Value(m) = 1, it means that the log tv does not exist before the data table, and the m operation occurs in the log record;

(2) If Value(m)=-1, it means that the log tv exists before the data table, and the m operation is deleted in the log record;

(3) If Value(m)=0, it means that the data table cannot determine whether the log tv exists. After the m operation, the log record still exists.

When the data to be exchanged is updated, the database operation log should be recorded. For each Sequence (T, v), calculate Value(Sequence (T, k)) .

According to the above log optimization method, by calculating the value of Value(Sequence (T, k)) , the user's multiple operations on tv can be combined, and only the last result can be used to avoid waste of resources. Improve the efficiency of data synchronization.

# 4. The Specific Realization of Log Optimization in the Observation Data Exchange Subsystem of the Station Network

## 4.1. Operation Log Table Design

The data synchronization in the earthquake precursor observation system has the data synchronization of the specified instrument, and during the incremental data synchronization of the specified instrument, the operation log of the specified instrument needs to be filtered out in the virtual operation log table, because the data operation log table must be filtered out[13-15]. Specifies the information fields for the instrument. According to this requirement, the operation log table structure required for precursor data exchange is shown in Table 1:

Table 1: Data exchange operation log table structure

| field name | Field meaning | type of data |
|---|---|---|
| RecordIndex | log number | Number (38) |
| TableName | data table name | V archar2(50) |
| Operation | Action type | V archar2(20) |
| UUID | Globally unique identifier for data | V archar2(255) |
| StationID | station number | V archar2(5) |
| PointID | serial number of the device | V archar(1) |

## 4.2. Pseudo-Code Representation of Operation Log Table Optimization Algorithm

I , U, D actions optimization is as follows:

```
Loop from begin currentLogId-maxId
    find current Data's pk
       if find identical log
          keep the current log
        endif
       else
   insert current Data's log insert update delete action
        end else
  end loop
```

## 4.3. Implementation of the Log Table Optimization Algorithm in the Precursor Project

In the precursor data management system, the data entry and exit operations are all realized through the unified data access platform, which realizes the CRUD operation of the data by providing distributed DAO components based on the data object view[16-17]. On the object data access interface of the DAO component, there is a flag of whether to exchange or not, and the exchange data is controlled by the invocation of the business layer. The code of the log optimization algorithm can be implemented in two ways. One is to record the virtual log through the interceptor when the data update enters the game, and realize the optimization of the virtual log during the log recording operation. The other is that data storage does not consider log optimization, and optimizes all virtual log sets that need to be synchronized at one time during data synchronization. According to the test results, there is little difference in performance between the two implementations, but the implementation code of the second implementation can be briefly introduced. The implementation code of some optimization algorithms is shown below.

```
List<Operation Log s> logs = dao.findAll();
LinkedHashMap<String, Operation Log s > map =
   new LinkedHashMap<String, Operation Log s >();
for(int i=logs.size();i>=1;i--){
   Operation Log s log = logs.get(i-1);
   if(!map.containsKey(log.getUuid())){
      map.put(log.getUuid(), log);
   }
}
List < Operation Log s > list = map.values();
Iterator< Operation Log s >it= list .iterator();
List<Operation Logs >temp= newArrayList<Operation Log s>();
while(it.hasNext()){
   Operation Log s log =it.next();
   temp.add(log);
}
for(int m=temp.size();m>=1; m--){
   sortList.add(temp.get(m-1));
}
```

## 4.4. Test Environment and Test Results

Access the entity objects of the platform through unified data, simulate 20,000 lines of precursory observation data, including 100 repeatedly collected data and 2,000 high-frequency modified preprocessing data, perform random save, update or delete operations every 20 microseconds, and record the results to the operation log. In order to ensure that the overall number of records in the test library will not change too much, the number of virtual log records of insert, update, and delete should be the same. Table 2 shows the average value of three virtual log optimization experiments.

Table 2: Experimental data

| Delete | Insert | Renew | Original number | Optimized |
|---|---|---|---|---|
| 0 | 0 | 2000 | 20000 | 16864 |
| 200 | 200 | 1600 | 20000 | 17746 |
| 400 | 400 | 1200 | 20000 | 18248 |

| | | | | |
|---|---|---|---|---|
| 600 | 600 | 800 | 20000 | 18512 |
| 800 | 800 | 400 | 20000 | 18842 |

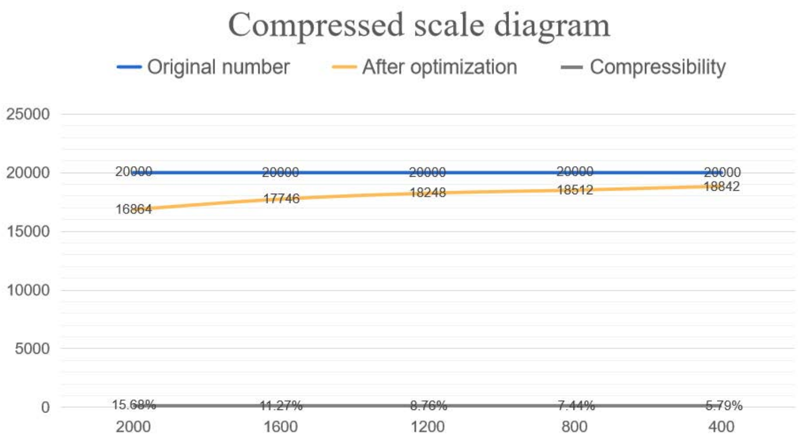


Fig.3: The relationship between the compression effect and the proportion of updates

The experimental results show that the preprocessing frequency of precursor data is highly linear with the optimization effect. With the increase of preprocessing frequency, the data exchange effect based on log compression is also improved. Log optimization can reduce exchange costs by about 15% for 2,000 preprocessing data out of 20,000 recorded precursor data. Even if there is no update operation among the 20,000 records, log optimization can reduce the switching cost by 5%, mainly by optimizing the synchronization cost caused by data re-collection.

## 5. Concluding Remarks

This paper designs and implements a data synchronization scheme based on log compression to improve the exchange reliability and synchronization efficiency of the precursor database, and obtain the expected efficiency. This solution is especially suitable for a distributed data business system environment with slow network speed, high update frequency of lower-level data sources, low data real-time requirements, and high data consistency requirements. For insurance, banking, aerospace and other data exchange scenarios with guaranteed network bandwidth, low network latency, and zero tolerance for data consistency, it is also possible to optimize the structure of the virtual log table, introduce transaction isolation level management, and adjust the log exchange frequency. Optimization, the optimization technology of this scheme has certain versatility and has certain promotion value.

## 6. Acknowledgements

National Key Research and Development Program: Strong Earthquake Monitoring Internet of things system development (2019YFC1509504).

## 7. References

[1] Zhang Chunling,Lv Zhenyu,Liu Zun.Database synchronization scheme based on virtual log compression[J].Computer Engineering,2010,36(18):0067-03.

[2] Ni Yongzhi,Wang Shan.A data replication solution based on virtual log [J].Computer Science,2007,34(3):78-82.

[3] Jin Songhe,Zhao Jinchao.Research on data synchronization system under distributed heterogeneous database[J].Journal of Yunnan University,2009,35(5):68-72.

[4] Zhang Hu,Dong Xiaoshe,Wu Weiguo,etc.A data synchronization mechanism based on log merge optimization[J].Small and Microcomputer Systems,2006,27(12):2183-2188.

[5] Zhang Zhen.Research and Implementation of Heterogeneous Database Synchronization System[J].Computer Applications;2002,(10).

[6] Yang Hongzhang,Luo Shengmei,Shi Jingchao,etc.Cloud storage system optimization for mobile communication big data [J].Computer Applications;2017, (S1):27-33.

[7] Pei Darong.The application of Web log mining technology in the optimization of e-commerce website[J].Science and Technology Economic Market;2009,(7):11-12.

[8] Ren Yan.Discussion on Oracle Database Performance Adjustment and Optimization Technology[J].China New Communication;2013,(1):3-4.

[9] Liu Chao,Zhang Mingan.Oracle database system optimization and adjustment[J].Information Security and Technology;2014,(7):101-104.

[10] Wang Yu.Research on performance optimization of Oracle database[J].Science and Technology Information;2009,(7):9.

[11] Sun Jiabo,Cai Peng.Query optimization technology of log-structured merge tree[J].Journal of East China Normal University; 2021,(9):94-103.

[12] Wu Shangyu,Xie Jingwen,Wang Yi.Log-structured merge tree optimization technology for key-value storage[J].Computer Research and Development;2020,(11):2432-2441.

[13] Liu Heyu,Zhang Li,Yang Xinghua,etc.Research on MapReduce performance optimization technology for DNS logs[J] . Intelligent Computer and Application;2018,(4):73-77.

[14] Liu Gaochuan,Teng Yuntian,Wang Chen,etc.The application of Oracle replication technology in the exchange of earthquake precursor metadata[J].China Earthquake;2008,(6):142-149.

[15] Yang Guanze,Shan Weifeng.Research on the visual similarity evaluation technology of earthquake precursor big data[J]. Journal of Disaster Prevention Science and Technology Institute;2020,(9):42-46.

[16] Chen Xiaolin,Li Shengle,Liu Jian,etc.Application of distributed database Greenplum in earthquake precursor data storage[J]. Earthquake Research;2020,(4):412-416+418.

[17] Wang Yuanyuan,Sun Keke,etc.Design and implementation of earthquake precursor data analysis algorithm based on LSTM [J].Intelligent Computer and Application;2020,(2):228-231.