Spatio-Temporal Association Query Algorithm for Massive Video Surveillance Data in Smart Campus

Jiwei Zhang
School of Computer Science, Beijing University of Posts and Telecommunications, Beijing, 100876, China
Corresponding author: Jiwei Zhang (e-mail: zhangjiwei5510@163.com).

ABSTRACT When using traditional methods to query massive video surveillance data in intelligent campus, there are some problems such as unstable query and inefficient query. Therefore, a query algorithm for intelligent campus video surveillance data based on spatio-temporal correlation is proposed. In this study, a spatio-temporal association query algorithm for massive video surveillance data in smart campus was proposed. First, the spatio-temporal data clustering algorithm was introduced to cluster the massive data of surveillance video in smart campus. Then, HBASE was used as the overall query structure of spatio-temporal association query algorithm based on the clustering results. Through combining the spatio-temporal features and attributed characteristics, a hierarchical record table was generated to construct the spatio-temporal attribute index of queries. According to the index of attribute columns, we can query massive data in many cases. The query condition was determined by Z curve, and the spatio-temporal association query of massive video surveillance data in smart campus was realized. Experimental results showed that when the number of data node was 5, the execution time of the algorithm of this paper was only 1200s, which was much shorter than the other traditional algorithms. It was proved that the algorithm can maintain spatio-temporal index, improve query efficiency and enhance query stability.

INDEX TERMS Smart Campus; surveillance video Massive data Spatio-temporal index Spatio-temporal association query

1. INTRODUCTION

Smart campus is proposed with the development of Internet technology, and normally used to realize sensing, perception, acquisition and transmission of massive data. At present, the smart campus has been widely applied in information communication [1]. When using the smart campus to exchange data, massive surveillance video data will be generated. Unlike traditional data, surveillance video data contain massive spatio-temporal characteristics. How to effectively store, manage and query big data by using these characteristics remains an urgent problem [2-4].

In order to improve the ability of the smart campus to process the massive surveillance video data, related scholars began to study the spatial and temporal association query method of the smart campus. Xu A P et al.[5] proposed a spatio-temporal association query algorithm based on Z curves. This algorithm can calculate the range of Z values in query space and set up space subsets, which can be used to effectively solve the defects of the time and complete space index construction and maintenance. However, this method has low query efficiency. Zhou F [6] proposed a mass data mining system of smart campus based on apriori algorithm of association rules. This method can quickly analyze, process, store and mine valuable information, but the query accuracy of this method is low. Xin J L[7] proposed a node selection algorithm for multi-objective tracking, which can be used to preprocess to the initial integer programming problem and turns into a convex optimization problem. By solving it, the sub optimal solution of node selection problem is obtained. However, the algorithm cannot adapt to the large amount of smart campus. Lou S S and Song C P[8] proposed a method of distributed reasoning and continuous query processing based on state migration, and compressed query processing state. According to the advantages and disadvantages of various architectures, a method of state migration by using RFID with storage medium was proposed in combination with the characteristics of RFID tag with its own storage medium. Finally, the above data cleaning & storage methods and complex event processing methods are combined to form the query of video surveillance data in the smart campus. This method has high accuracy, but poor stability.

To solve the existing problems, a spatio-temporal association query algorithm for massive video surveillance data in smart campus was proposed. This paper analyzed the characteristics and research significance of massive
monitoring video data in the smart campus. In order to realize spatio-temporal query, the massive monitoring video data in the smart campus were clustered; A HBASE based storage model and index scheme based on the clustering results were established, and the HBASE manager index scheme was used to construct large monitoring video data. Finally, the spatio-temporal correlation query of video surveillance data was achieved. The experimental results show that this method is efficient and stable in querying massive surveillance video data of the smart campus.

2. CHARACTERISTICS AND RESEARCH SIGNIFICANCE OF MASS SURVEILLANCE VIDEO DATA IN SMART CAMPUS

In the process of wide application of smart campus, massive surveillance video data will be generated. These booming surveillance video data contain rich information. By fully developing and using these information, the development and application of the smart campus technology can be promoted. Relevant experts and scholars have studied the massive surveillance video data in smart campus. It is found that these massive surveillance data have the following characteristics.

First, the number of data is huge. The basic equipment in smart campus is a sensing device, and each sensor device consists of a surveillance device or a surveillance module, in which the number of surveillance video data is up to 10000. Secondly, the type of surveillance video data is more complex. This is mainly due to that surveillance objects involved in the smart campus system are various, and the surveillance video information collected by different surveillance object is not the same. Thirdly, massive surveillance video data are heterogeneous. There are a large number of sensing devices in smart campus system, that is, sensing terminals as well as the format and semantics of surveillance video data collected by different sensor terminals are different. This results in heterogeneity of massive monitoring video data and increases the difficulty of data processing. Due to the three features mentioned above, rapid and accurate query of massive monitoring video data has become a big challenge. Therefore, a spatio-temporal association query algorithm for massive surveillance video data the smart campus was proposed in this study.

3. HBASE BASED SPATIO-TEMPORAL ASSOCIATION QUERY ALGORITHM FOR SURVEILLANCE VIDEO DATA IN SMART CAMPUS

3.1 STORAGE MODEL AND INDEX SCHEME BASED ON HBASE

HBase is a column-oriented, versioned and scalable distributed database system. HBase adopts a Bigtable storage model similar to Google, the bottom of which is a sparse, distributed, persistent storage multidimensional sort Map structure that supports fast query of single records, the addition and deletion of any specified location single or continuous batch data, and conditional query of the column index level. These characteristics make it very suitable for dealing with massive spatio-temporal data. The following is a comparative analysis of HBase related column search solutions.

1) IHBBase scheme.

The core idea of IHBBase (indexed HBase) is to preserve corresponding relation of the index by using the format of bit mapping. With small occupancy space, high query efficiency, but low update efficiency, it is more suitable for read-only data.

2) ITHBase scheme.

The core idea of ITHBase (indexed transactional HBase) is to separately store index data in a HBase table. When the client initiates the write operation, it first updates the index value of record to index table, and then writes data to HBase table. However, the whole process needs to launch two RPC requests to the server, which will inevitably increase operation delay. It is very difficult to maintain data consistency when updating and deleting data.

3) Coprocessor indexing scheme.

The core idea of coprocessor index is to ensure that the index and main table are on the same Region Server. When the index or query is set up, only a connection with the Region Server is needed. The main server reads the configuration file via index coprocessor to collect the information and build the index; in contrast, the partition server Using callback function to create and manage index data, so it is simple and easy to carry out index maintenance and guarantee data consistency.

3.2 DESIGN OF CLUSTERING ALGORITHM FOR SPATIO-TEMPORAL DATA

In spatio-temporal dimension, the spatio-temporal neighborhood entities of massive video data in smart campus have an adjacency relation. At the same time, due to the diversity of big data itself, the spatio-temporal data of big data should be clustered firstly to realize spatio-temporal queries. The clustering process is as follows:

1) First, a spatio-temporal entity is selected as the spatio-temporal center. If all the spatio-temporal entities in the time neighborhood and the spatial domain satisfy spatio-temporal adjacency conditions, then the spatio-temporal entity is considered as the initial spatio-temporal center.

2) Taking the initial spatio-temporal center as the core, the spatio-temporal neighborhood defined in the previous definition is used to judge the near distance relationship between the spatio-temporal entities and the spatio-temporal
center. Then, the nearest spatio-temporal entity is added in clustering set, and the first cluster set is generated.

(3) The cluster set is expanded according to the steps (2). The spatio-temporal entity that has been added to the clustering set is used as the extension center to further judge the surrounding entities. In turn, the entities that meet the spatio-temporal adjacency conditions are added to the clustering set until the surrounding entities are not conformed to the conditions. At this time, a cluster set is completed.

(4) Judging the remaining spatio-temporal entities that have not been clustered. If one is not marked as an isolated point, it can be regarded as another initial spatio-temporal center. Repeating the operation of step (1) to (3), if all spatio-temporal entities belong to a cluster set, or are labeled as isolated points, the whole clustering calculation is completed.

3.3 CONSTRUCTION OF SURVEILLANCE VIDEO DATA INDEX IN SMART CAMPUS

In order to efficiently store and query massive surveillance video data, this paper designs an index building method based on HBase and spatio-temporal correlation dependency rules. First, the spatio-temporal features of surveillance video data and the attribute characteristics with interdependence were excavated. Then, the associated eigenvalues for rowkey were constructed using Z curve. Finally, the paper describes the construction of surveillance video large data index process by the HBase coprocessor index scheme in detail.

3.3.1 ASSOCIATION AND DEPENDENCY RULES FOR SURVEILLANCE VIDEO LARGE DATA

<table>
<thead>
<tr>
<th>Data object</th>
<th>Dependent eigenvalues</th>
<th>Spatio-temporal association value</th>
<th>File path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Record1</td>
<td>camera1, case1,</td>
<td>T1, G1( x1, y1)</td>
<td>filepath1</td>
</tr>
<tr>
<td>Record2</td>
<td>camera2, case2,</td>
<td>T2, G2( x2, y2)</td>
<td>filepath2</td>
</tr>
<tr>
<td>Record3</td>
<td>camera3, case3,</td>
<td>T3, G3( x3, y3)</td>
<td>filepath3</td>
</tr>
</tbody>
</table>

3.3.2 ROWKEY DESIGN OF MASSIVE SURVEILLANCE VIDEO DATA

On the basis of determining the feature association and dependency rules of massive surveillance video data, it can be known that the index healthy rowkey of HBase is composed of time feature value Time, spatial eigenvalue Geo (X, Y), attribute eigenvalue Properties[ camera, case, record]. The time characteristic value is used as the time characteristic part of rowkey according to the Green time format; the attribute eigenvalues are separated by filling bits, and then spliced into a character feature part of rowkey; the eigenvalue of vector space uses the Z curve to reduce the dimension and compacts the two-dimensional vector eigenvalue Time (X, Y). One dimensional Z-order value is part of rowkey, Z curve is a simple but efficient space filling curve, which brings together space objects with similar coordinates, and has a good spatial continuity on the whole. The final rowkey is:

\[
\text{rowkey} = \text{Properties} + \text{Z-order} + \text{T} \quad (1)
\]

3.3.3 DESIGN OF CLUSTER OF MASSIVE SURVEILLANCE VIDEO DATA

In this section, the HBase storage model is used to design spatio-temporal eigenvalues and attribute eigenvalues into HBase column clusters. The time feature is stored with 14 bit time placeholders, the vector space point is stored in two dimensional coordinates in GeoTools, and the vector space point transverse ordinate is parsed into a string type in HBase. Temporal and spatial data together constitute the space-time cluster TGeo. Attribute features are stored in Properties column clusters and stored in string format. The name of the column attribute is event number, probe number and record number, each of which occupies 4 place. The attribute names in each cluster are dynamically increased as column qualifiers of HBase, enabling cluster mode to be reused and expanded. The design of cluster for massive surveillance video data is shown in Table 2.

<table>
<thead>
<tr>
<th>RowKey</th>
<th>Column Family: Qualifier</th>
<th>Properties</th>
<th>Column Family: Qualifier</th>
<th>TGeo</th>
</tr>
</thead>
<tbody>
<tr>
<td>camera</td>
<td>Camera id</td>
<td>GeoZ</td>
<td>Z-order</td>
<td></td>
</tr>
<tr>
<td>case</td>
<td>Case id</td>
<td>GeoX</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>record</td>
<td>Record id</td>
<td>GeoY</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>file</td>
<td>File path</td>
<td>Time</td>
<td>T</td>
<td></td>
</tr>
</tbody>
</table>
3.3.4 CONSTRUCTION OF MASSIVE SURVEILLANCE VIDEO DATA INDEX

On the basis of clustering results, HBase is used as a spatiotemporal correlator to realize efficient query of massive spatio-temporal video data in a smart campus. In this paper, the query structure of massive surveillance video data was studied. According to the storage structure and generation rule of rowkey of characteristic values of surveillance video big data, the spatio-temporal characteristics and attribute characteristics were used to generate the HBase Table records and construct the secondary index of attribute columns, which provides the basis for subsequent spatio-temporal association query of big data. The construction process of the index is shown in Figure 1.

For each big data file of surveillance video, firstly, the Z curve is used to encode the spatial features corresponding to the file into one-dimension Z-order value, and a single row is set in accordance with the rowkey generation function and inserted into the first level index table HBase Table. Then, the coprocessor is used to pre-write the interface preFlush to establish spatio-temporal index. The purpose of this is to support conditional query in three dimensions: time column, space column and attribute column. The next step is to determine whether the column index needs to be set up as the secondary index [9]. If it is not, the RPC connection needs to be disconnected; otherwise, the Indexmanager is created to find the data that needs to be indexed through the pre-read interface preFlush. After obtaining the data in memory, firstly, Key and Value are turned over. The column values are used as Key, and rowkey is used as Value. Then, the reconstructed key value pairs are regarded as index values to be stored in the IndexTable of the col:qualifier index directory. Finally, the corresponding column identifiers are saved into the data structure of IndexMessager, and the specific location of index file information saved in IndexFile is specified [10].

3.4 DESIGN OF SPATIO-TEMPORAL ASSOCIATION QUERY ALGORITHM FOR SURVEILLANCE VIDEO DATA IN SMART CAMPUS

On the basis of above, the secondary index and column key are used to query the spatio-temporal association characteristics of massive surveillance video data [11]. The query request is marked as \( Q(Q,G,Q,P,Q,T) \). Where, \( Q,G \) represents spatial characteristics, \( Q,P \) represents attribute characteristics, and \( Q,T \) represents time characteristics, and all records satisfying the query condition \( (Q,G,Q,P,Q,T) \) are found. For this problem, this paper considers two aspects of index query analysis and algorithm strategy, and proposes two spatio-temporal query algorithms.

3.4.1 SPATIO-TEMPORAL ASSOCIATION QUERY ANALYSIS BASED ON COLUMN INDEX

In order to effectively support multi-conditional query of spatio-temporal characteristics of surveillance video data, for a given query request \( Q(Q,G,Q,P,Q,T) \), the query requests are classified and calculated according to the default of the eigenvalues in the query conditions. A total of seven queried states of them are obtained: \( TGP; TP; TG; GP; T; G \) and \( P \). The calculation process is as follows:

Figure 1. Construction process of massive surveillance video data index
When there is no default in the query condition, a query state is obtained: \( TP \); When there are a set of eigenvalues missing in the query condition, three query states are obtained: \( TP, TG, GP \); When there are two sets of eigenvalues missing in the query condition, \( T, G \) and \( P \) are obtained. For the seven query states that may exist in \( Q \left( Q.G, Q.P, Q.T \right) \), three index candidate keys of \( TP, TG \) and \( GP \) are obtained. Considering sequential problems, six index schemes are obtained via combination arrangement. The composition of rowkey is \( PTG, PGT, GTP, GPT, TGP, TPG \), where \( TPG \) matches the rowkey rules. Therefore, the establishment of five index tables can satisfy all the query requirements [12].

### 3.4.2 ANALYSIS OF SPATIO-TEMPORAL ASSOCIATION QUERY BASED ON COLUMN INDEX

In order to effectively support multi-conditional query of spatiotemporal feature values of surveillance video data, for a given association query request \( Q \left( Q.G, Q.P, Q.T \right) \), the query requests are classified according to the default of each characteristic value in the query condition, and seven kinds of query states are obtained: TGB, TP, TG, GP, T, G, P. The calculation process is as follows:

A query state is obtained when the eigenvalues are not absent in the query condition: TGP; three query states are obtained when the eigenvalues of the query condition are missing in a set of eigenvalues: TP, TG, and GP; when the eigenvalues of the query condition are missing, two groups of eigenvalues are missing, then T, G, P are obtained.

For \( Q \left( Q.G, Q.P, Q.T \right) \), there are seven kinds of query states to build index, and three groups of index candidate keys \( TG, TP, GP \) can be obtained. Considering sequential problems, six indexing schemes are obtained through combination arrangement. The composition of rowkey is \( PTG, PGT, GTP, GPT, TGP, TPG \), among which \( TPG \) exactly is the rowkey in the type (1). Therefore, the establishment of five index tables can satisfy all the query requirements.

### 3.4.3 SPATIO-TEMPORAL ASSOCIATION QUERY ALGORITHM BASED ON Z CURVE

On the basis of the above multi-conditional query, ZRMF algorithm (Z-order range with MultiFilter based algorithm) is used to realize accurate query of spatio-temporal association of massive surveillance video data. The specific steps are shown in algorithm 1.

Algorithm 1: ZRMF based spatio-temporal association query algorithm

**Input:** spatio-temporal association word query \( Q \left( Q.G, Q.P, Q.T \right) \), of which \( Q.G \) is a rectangular spatial feature constructed by the vertex coordinate \( leftLow \) in the lower left quarter and the vertex coordinate \( rightHigh \) in the upper right corner [13]. \( Q.P \) is a set of attribute characteristics for string splicing, and \( Q.T \) represents the time feature range.

**Output:** the result set list \( RecordList \)

1. \( Z_{\min} \leftarrow calculateMinZorder(\text{leftLow}) \)
2. \( Z_{\max} \leftarrow calculateMaxZorder(\text{rightHigh}) \)
3. \( Z\text{Ranger} \leftarrow calculateZorderRange(\left( Z_{\min}, Z_{\max} \right)) \)
4. \( P\text{Range} \leftarrow calculatePRange(P_{\min}, P_{\max}) \)
5. If \( \left( P\text{Range}, Z\text{Range} \right) \) is unique
6. \( \text{rowkeyRanger} \leftarrow \text{LeftShift}\left( Q.P + Q.Z \right) \)
7. \( \text{IndexP} \leftarrow \text{IndexManager}(P\text{Range}) \)
8. \( \text{IndexG} \leftarrow \text{IndexManager}(Z\text{Range}) \)
9. \( \text{CandidateP} \leftarrow \text{IndexTableP.\text{preScan}}(\text{IndexP}) \)
10. \( \text{CandidateG} \leftarrow \text{IndexTableG.\text{preScan}}(\text{IndexG}) \)
11. \( \text{rowkeyRanger} \leftarrow \text{CandidateP} \cap \text{CandidateG} \)
12. \( \text{scanner} \leftarrow \text{Scanner}\left( \text{rowkeyRanger}, \text{bloomFilter}(T,G) \right) \)
13. \( \text{RecordList} \leftarrow \text{HBaseTable.\text{getScan}}(\text{scanner}) \)

In step (a), the Z-order value of the lower left point in the rectangle space is calculated, and \( Z_{\min} \cdot b \) is obtained. The Z-order value \( Z_{\min} \cdot e \) of the top right point in the rectangular space is calculated by the Z-order value. From \( Z_{\min} \text{ and } Z_{\max} \), the value range \( Z\text{Range} \cdot d \) of 3 after dimension reduction is obtained, and the value range \( P\text{Range} \cdot e \) of \( P \) is obtained from parameter \( P \). The step is to determine whether \( P\text{Range} \) and \( Z\text{Range} \) are unique or not. If they are, \( f \) is executed. \( PZ \) is used to splice and construct \( \text{rowkey} \) and move left to form the lookup scope \( \text{rowkeyRange} \); otherwise turn to \( g \) \( \sim k \). According to \( P\text{Range} \) and \( Z\text{Range} \), index condition objects \( \text{IndexP} \)
and IndexZ are constructed. Then, index tables IndexTableP and IndexTableG are selected by using index objects IndexP and IndexZ. The hook function in column index scheme perScan is invoked to scan index table [14], and then candidate set CandidateP and CandidateZ which satisfying index condition are obtained. Finally, the candidate sets CandidateP and CandidateZ are intersected, and the search range rowkeyRange;1(m) of column key satisfying both Q.P and Q.Z conditions is obtained. Using rowkeyRange, parameters Q.T and Q.P, scanning object scanner is constructed to scan the original table of HBase, obtain result set list RecordList and return to client [15].

The ZRMF algorithm can achieve very good results for the case with good spatial continuity (i.e. with high density of Q.G coordinate points) [16]. However, when the continuity of the Q.G space is not good, there will be more useless candidate column key in Z.Range. Especially when the query space range Q.G increases and the traffic volume of invalid Z-order to HBase is greater, the efficiency of the ZRMF algorithm will be reduced. Therefore, a spatio-temporal association query algorithm based on the ZRMF algorithm is proposed. The algorithm uses KD-tree to query space range and divide it into smaller space blocks [17], and construct the index range kd1 on the partition block. When querying, Z.Range and kd are intersecting, and the invalid Z-order values are filtered out to form a more accurate Z.Range after each partition. The following algorithm process is the same as that of ZRMF, as shown in algorithm 2.

Algorithm 2: spatio-temporal association query algorithm based on kd-ZRMF

Input: spatio-temporal association word query \( Q(Q.G, Q.P, Q.T) \), of which Q.G is a rectangular spatial feature constructed by the vertex coordinate leftLow in the lower left quarter and the vertex coordinate rightHigh in the upper right corner. Q.P is a set of attribute characteristics for string splicing, and Q.T represents the time feature range.

Output: the result set list RecordList
(a) \( Z_{\text{min}} \leftarrow \text{calculateMinZorder}(\text{leftLow}) \)
(b) \( Z_{\text{max}} \leftarrow \text{calculateMaxZorder}(\text{rightHigh}) \)
(c) Z.Range \leftarrow \text{calculateZorderRange}(Z_{\text{min}}, Z_{\text{max}})
(d) kd-tree \leftarrow \text{buildKdTree}(Q.G)
(e) for \( kd_1 \) in kd-tree do
   Z.Range, \leftarrow Z.Range \cap kd_1
   Z.RangeSet.add(Z.Range)
end for
(f) P.Range \leftarrow \text{calculatePRange}(P_{\text{min}}, P_{\text{max}})
(g) if (P.Range, Z.Range unique)
(h) rowkeyRange \leftarrow \text{LeftShift}(Q.p + Q.Z) else
(i) IndexP \leftarrow \text{IndexManager}(P.Range)
(j) CandidateP \leftarrow \text{IndexTableP.preScan}(IndexP)
(k) for Z.Range, in Z.RangeSet
   IndexZ \leftarrow \text{IndexManager}(Z.Range)
   CandidateG \leftarrow \text{IndexTableG.preScan}(IndexZ)
   rowkeyRange, \leftarrow CandidateP \cap CandidateZ
   rowkeyRange.add(rowkeyRange,)
end for
(l) scanner \leftarrow \text{Scanner}(rowkeyRange, colFilter(T))
(m) RecordList \leftarrow \text{HBaseTable.getScan}(scanner)

In the algorithm, step (a) - (c) is consistent with algorithm 1, and the value range Z.Range of the spatial characteristic Q.G after dimension reduction is calculated. (d) the query space determined by parameter Q.G is divided by using the KD-tree [18], to obtain the division result kd-tree, kd-tree, which is the K-Value structure composed of the Z-order values in the sub block and the sub block range. (e) each division sub-block \( kd_1 \) in kd-tree is traversed, and divided by spatial continuity conditions, so the discontinuous invalid Z-order is divided into different sub blocks to filter more conveniently.

Then, Z-order and kd are intersected to obtain the exact range Z.Range, of Z.Range on each space sub-division.
block $kd_i$. $ZRange_i$ obtained in each cycle is added to $ZRangeSet$. (f) the value range $PRange$ of $P$ is calculated; (g) to determine whether $PRange$ and $ZRange$ are unique. If they are, (h) will be executed, and $PZ$ will be used to splice and construct rowkey and move left to form the search scope of rowkey $Range$. Otherwise, turn to (i) - (k). according to $PRange$, the index condition object $IndexP$ is constructed, the corresponding index table $IndexTableP$ is selected, and the index table is scanned using the hook function $preScan$ to obtain a candidate set $CandidateP$ that satisfies the index condition [19-23]. Then, $ZRangeSet$ is traversed to scan the index table $preScan$ by the constructed index condition object $IndexZ$ of $ZRange_i$, and column index scheme $IndexZ$, and the candidate set $CandidateG$ satisfying $IndexZ$ can be obtained. Finally, the candidate sets $CandidateP$ and $CandidateG$ are intersected, and the search range rowkey$Range_i$ of column key satisfying both $Q\cdot P$ and $Q\cdot Z$ conditions is obtained. The obtained rowkey$Range_i$ of each cycle is added to rowkey$Range$. In (l) and (m), rowkey$Range$ and query parameter $QT$ are used to construct scanned object, scan the original table $HBase$ and obtain result set list $RecordList$.

4. EXPERIMENTAL ANALYSIS

In order to verify the validity of spatio-temporal association query algorithm for smart campus massive video surveillance data, the experiment will compare the difference between the traditional algorithm and this algorithm in query efficiency, query stability and query accuracy.

In order to test the effect of data amount on the algorithm, 500 thousand, 1 million, 2 million, 4 million, 8 million videos were processed by the proposed algorithm, Par2PK-Means algorithm and ParCLARA algorithm, respectively. The query response time is shown in Figure 2.

From Figure 2, it can be seen that the three algorithms can perform better query efficiency when the amount of data is small. With the increase of data amount, the query performance of Par2PK-Means algorithm is seriously reduced. The response time curve of the proposed algorithm is stable, which shows that the change of data amount has little effect on the performance of the proposed algorithm.

In order to test the effect of attribute feature set on the algorithm, three different algorithms including the proposed algorithm were tested based on the discrete attribute feature sequence. The results of the query experiment are shown in Figure 3.

From Figure 3, we can see that the advantage of Par2PK-Means algorithm in searching discrete attribute eigenvalues is weakened. In contrast, the proposed algorithm can solve the interference of the discrete attribute sequence, and remain stable query performance [24-27].
In order to test the index insertion speed of the proposed algorithm, the proposed algorithm and the ZRMF algorithm were compared in terms of index insertion speed. The result is shown in Figure 4.

As shown in Figure 4, the insertion speed of the proposed algorithm is 2-3 times faster than that of the ZRMF algorithm under different data amounts, and the insertion speed of the ZRMF algorithm is obviously reduced as the amount of data increases, while the insertion speed of the proposed algorithm is stable. The experimental results show that data amount has little effect on the insertion speed of the proposed algorithm.

In order to verify the acceleration performance of the proposed algorithm, the proposed algorithm, MR-PFP algorithm and PFP algorithm were analyzed and compared in the same environment. Firstly, the acceleration ratio performance of the proposed algorithm was evaluated in the case that the sizes of the three real data sets (Dataset 1, Dataset 2 and Dataset 3) are the same, and the experimental results are shown in Figure 5. Secondly, the acceleration ratio performances of the three algorithms were compared, and the results are shown in Table 3.

![Figure 4. Comparison of index insertion speed under different data amounts](image-url)

**Figure 4. Comparison of index insertion speed under different data amounts**

<table>
<thead>
<tr>
<th>Data amount (ten thousand)</th>
<th>ZRMF algorithm</th>
<th>Algorithm in this paper</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>6000</td>
<td>5000</td>
</tr>
<tr>
<td>100</td>
<td>5000</td>
<td>4000</td>
</tr>
<tr>
<td>200</td>
<td>4000</td>
<td>3000</td>
</tr>
<tr>
<td>400</td>
<td>3000</td>
<td>2000</td>
</tr>
<tr>
<td>800</td>
<td>2000</td>
<td>1000</td>
</tr>
</tbody>
</table>

**Figure 5. The acceleration ratios of different algorithms**

<table>
<thead>
<tr>
<th>The number of datanodes/num.</th>
<th>Running times/S</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1800</td>
</tr>
<tr>
<td>2</td>
<td>1600</td>
</tr>
<tr>
<td>3</td>
<td>1400</td>
</tr>
<tr>
<td>4</td>
<td>1200</td>
</tr>
</tbody>
</table>

**TABLE 3**

<table>
<thead>
<tr>
<th>Datanode</th>
<th>Dataset</th>
<th>Algorithm</th>
<th>Running time/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Datane</td>
<td>Dataset 1</td>
<td>MR-PFP algorithm</td>
<td>768</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PFP algorithm</td>
<td>824</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Algorithm in this paper</td>
<td>1082</td>
</tr>
<tr>
<td></td>
<td>Dataset 2</td>
<td>MR-PFP algorithm</td>
<td>1322</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PFP algorithm</td>
<td>1407</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Algorithm in this paper</td>
<td>1790</td>
</tr>
<tr>
<td></td>
<td>Dataset 3</td>
<td>MR-PFP algorithm</td>
<td>2288</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PFP algorithm</td>
<td>2369</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Algorithm in this paper</td>
<td>371</td>
</tr>
<tr>
<td></td>
<td>Dataset 1</td>
<td>MR-PFP algorithm</td>
<td>614</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PFP algorithm</td>
<td>682</td>
</tr>
<tr>
<td>2 Datane</td>
<td>Dataset 2</td>
<td>MR-PFP algorithm</td>
<td>1225</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PFP algorithm</td>
<td>1306</td>
</tr>
<tr>
<td></td>
<td>Dataset 3</td>
<td>Algorithm in this paper</td>
<td>1573</td>
</tr>
</tbody>
</table>
It can be seen from Figure 5 that when mining frequent item sets from data sets consisting of large amounts of small files, the proposed algorithm can complete the mining tasks normally. Especially, when the Datanode gradually increases, the running time of the proposed algorithm is reduced proportionally. As can be seen from Table 1, when the Hadoop cluster is transitioning from a pseudo distributed environment (only one Datanode in the cluster is involved in computing) to a fully distributed environment (with two or more Datanodes involved in the calculation), the processing capability of proposed algorithm is significantly improved. Especially when the Datanode increases gradually, the execution time of the proposed algorithm is much lower than the other two algorithms. The results show that the proposed algorithm has better acceleration ratio performance and can effectively improve the processing efficiency.

In order to verify the memory consumption of proposed algorithm, three different algorithms were used to calculate the Namenode memory space of five sets of data sets (from Dataset1 to Dataset5). The result is shown in Figure 6.

![Figure 6. Comparison of memory consumption of different algorithms](image)

Based on the experimental results in Figure 7, it can be known that when the number of datanode is 5, the execution time of SF algorithm, HAR algorithm and the algorithm of this paper reaches the maximum, but the execution time of the algorithm of this paper is only 1200s, which is much shorter than the two traditional algorithms. The SF algorithm reduces the memory consumption of Namenode and shortens the scheduling time of HDFS, but the time recorded in the experiment is the execution time of the whole process of mining tasks, that is, the processing time of non-mining tasks are also included (such as a large amount of non-mining tasks consumed in the process of merging small files into a sequence file, and the execution time is counted as the running time of the whole task), thus real running time of SF algorithm is longer. In this paper, a large number of small files were packaged into a single nputSplit before the execution of MapReduce tasks. They are not merged into sequential files, and less memory is consumed. Experiments show that the memory consumption of the proposed algorithm is lower when processing massive surveillance video data.

In order to verify the efficiency of proposed algorithm, the execution times of the five data sets in the MapReduce environment under the proposed algorithm, HAR algorithm and SF algorithm were compared, and the experimental results are shown in Figure 7.

![Figure 7. Comparison of execution efficiency of different algorithms](image)

Through the analysis of Figure 6, we can know that when HAR algorithm is adopted, a large number of small files are archived into a*.har file, and there are two indexes of MasterIndex and Index coexisting in Namenode, so the memory space consumption of Namenode depends on the number of two indexes and the memory consumption is large. When the SF algorithm is used, a large number of small files are merged into a large sequence file, and an Index exists in the Namenode. The consumed memory space of Namenode depends on the number of the index, a large amount of memory is consumed. When the proposed algorithm is adopted, a large number of small files are packaged into nputSplit before the execution of MapReduce tasks. They are not merged into sequential files, and less memory is consumed. Experiments show that the memory consumption of the proposed algorithm is the best.
In order to detect the accuracy of the proposed algorithm, the proposed algorithm, Par2PK-Means algorithm, ParCLARA algorithm and the K mean algorithm were tested in the Hadoop colony consisting of eight nodes, and different sizes of data sets were clustered. The experimental results are shown in Table 4.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Algorithm in this paper</th>
<th>Error Rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iris</td>
<td>Algorithm in this paper</td>
<td>6.21</td>
</tr>
<tr>
<td></td>
<td>Par2PK-Means algorithm</td>
<td>7.38</td>
</tr>
<tr>
<td></td>
<td>ParCLARA algorithm</td>
<td>14.27</td>
</tr>
<tr>
<td></td>
<td>K-Means algorithm</td>
<td>11.06</td>
</tr>
<tr>
<td></td>
<td>Algorithm in this paper</td>
<td>15.22</td>
</tr>
<tr>
<td>Haberman's Survival</td>
<td>Par2PK-Means algorithm</td>
<td>16.38</td>
</tr>
<tr>
<td></td>
<td>ParCLARA algorithm</td>
<td>18.74</td>
</tr>
<tr>
<td></td>
<td>K-Means algorithm</td>
<td>15.31</td>
</tr>
<tr>
<td></td>
<td>Algorithm in this paper</td>
<td>17.75</td>
</tr>
<tr>
<td></td>
<td>Par2PK-Means algorithm</td>
<td>21.28</td>
</tr>
<tr>
<td>Ecoli</td>
<td>ParCLARA algorithm</td>
<td>20.86</td>
</tr>
<tr>
<td></td>
<td>K-Means algorithm</td>
<td>21.33</td>
</tr>
<tr>
<td></td>
<td>Algorithm in this paper</td>
<td>36.34</td>
</tr>
<tr>
<td></td>
<td>Par2PK-Means algorithm</td>
<td>37.89</td>
</tr>
<tr>
<td>Lenses</td>
<td>ParCLARA algorithm</td>
<td>42.57</td>
</tr>
<tr>
<td></td>
<td>K-Means algorithm</td>
<td>36.38</td>
</tr>
<tr>
<td></td>
<td>Algorithm in this paper</td>
<td>26.10</td>
</tr>
<tr>
<td></td>
<td>Par2PK-Means algorithm</td>
<td>29.09</td>
</tr>
<tr>
<td></td>
<td>ParCLARA algorithm</td>
<td>31.17</td>
</tr>
<tr>
<td></td>
<td>K-Means algorithm</td>
<td>26.09</td>
</tr>
<tr>
<td></td>
<td>Algorithm in this paper</td>
<td>22.50</td>
</tr>
<tr>
<td></td>
<td>Par2PK-Means algorithm</td>
<td>29.13</td>
</tr>
<tr>
<td></td>
<td>ParCLARA algorithm</td>
<td>30.55</td>
</tr>
<tr>
<td></td>
<td>K-Means algorithm</td>
<td>32.71</td>
</tr>
</tbody>
</table>

As can be seen from Table 4, the proposed algorithm produces more accurate clustering results than the other three algorithm in most cases.

In order to verify the extensibility of the proposed algorithm, the proposed algorithm, Par2PK-Means algorithm and ParCLARA algorithm were compared under the condition that the size of the dataset (from 100MB to 1280MB) and the number of nodes (from one node to eight nodes) are increased with the same proportion, and the experimental results are shown in Figure 8 (a). In addition, the Iris data set (from 160MB to 1280MB) was used to further verify the high scalability of the proposed algorithm by comparing with the Par2PK-Means algorithm and ParCLARA algorithm. The experimental results are shown in Figure 8 (b).

From Figure 8 (a), it is found that the expansion value of the proposed algorithm is less than 1 as the number of nodes increases proportionally to the size of the dataset. Specifically, the value of its expansibility is higher than 0.62. In addition, it is observed from figure 8 (b) that the proposed algorithm has better scalability than the Par2PK-Means algorithm and the ParCLARA algorithm (the scalability values of the proposed algorithm, the Par2PK-Means algorithm and the ParCLARA algorithm are derived from the scalability values of the proposed algorithm).
algorithm and the ParCLARA algorithm are higher than 0.66, 0.59 and 0.53 respectively). The results show that the proposed algorithm has excellent scalability and adaptability in processing large scale data sets on the Hadoop platform based on MapReduce framework.

The stability of the proposed algorithm was evaluated by gradually closing the node (from one node to seven nodes), and the same clustering results were obtained from the Iris data set (1280MB). The results of the experiment are shown in Figure 9.

![Figure 9](image)

Figure 9. The stability of the proposed algorithm

As shown in Figure 9, although the running time of the proposed algorithm grows with the increase of the number of node closed, it is still able to perform well and produce the same clustering results. The above results show that the algorithm has high stability under the "big data" environment.

5. CONCLUSIONS

In order to solve the disadvantages of inefficient query and unstable query under traditional methods, a spatio-temporal association query algorithm for massive surveillance video data based on HBase was proposed. The clustering algorithm of spatio-temporal data was used to cluster the massive surveillance video data and improve the querying efficiency. Based on the result of clustering distance, the multi-condition query of big data’s spatio-temporal correlation was realized by using the spatio-temporal association query analysis algorithm of the column index after modeling the big data index of surveillance video. Then, the spatio-temporal association query algorithm based on Z curve was used to complete the accurate query of spatio-temporal association. The experimental results fully verifies that this algorithm has high accuracy, efficiency and stability when querying spatio-temporal association of massive surveillance video data. The proposed algorithm has high query performance and achieves satisfactory results.

CONFLICTS OF INTEREST

The authors declare no conflicts of interest regarding the publication of this article.

REFERENCES

[10] Ellingsen V M, Asplund J, Ohlson M. Spatial and temporal patterns in seed predation as revealed by reciprocal experiments and video


Jiwei Zhang received a Bachelor’s degree from Yantai University in 2013, and Master’s degree from Beijing University of Posts and Telecommunications from 2013 to 2015, and a Ph.D. student in Beijing University of Posts and Telecommunications from 2015 till now. He has a strong interest in artificial intelligence and image forensics technology.