Web Intelligence 18 (2020) 133–143
DOI 10.3233/WEB-200435
IOS Press

Research on distributed data stream mining algorithms based on matrix weighted association rules

Dong Xu *
School of Communication Engineering, Chongqing College of Electronic Engineering, Chongqing, 401331, China

Abstract. In order to overcome the low efficiency of traditional data mining algorithms without considering weighted association rules, this paper proposes a distributed data flow mining algorithm based on matrix weighted association rules. According to the way of separating metadata and data flow, garbage data processing in data flow is realized. By using sliding window and data summary structure to optimize PCA algorithm, the main component decision matrix is formed in the window, and the dimension of data in sliding window is reduced by using the decision matrix. The matrix weighted association rules are used to mine the distributed data. After dimensionality reduction, the transactions in the database are clustered according to the time distribution. The weighted analysis is carried out for each aggregation to obtain the weighted frequent item set with time and output the mining results. The experimental results show that the proposed algorithm has high efficiency and the highest accuracy of 98.9%.

Keywords: Matrix weighted association rules, data flow, mining, data dimension reduction

1. Introduction

Since the 1960s, database technology has been widely used in government agencies and business sectors. In the context of the continuous development of the times, especially the emergence and popularization of Internet technology, the data in these areas continue to increase in an explosive way, which makes it very urgent for data sets to carry out some information extraction operations [4,6,11]. Current database technology has some limitations, such as database retrieval and query, which cannot obtain knowledge in the database, resulting in the rich knowledge contained in the database cannot be efficiently applied and excavated. Under this background, data mining technology emerges at the historic moment. The main function of data mining is to excavate hidden rules in massive data centers, so as to provide efficient support for decision-making and data utilization.

Data mining technology is relatively late in domestic research, and domestic data mining work is still staying in theoretical research. The relevant departments involved in research and development are mainly government departments, universities and IT enterprises with relatively large scale in China [7,15, 17]. In such an environment, the research and application of data mining has attracted the attention of domestic academia and industry. Through the combination of professional training, state funding and enterprise research, we can foresee that domestic data mining has very good prospects.

Data mining arises under the technology of knowledge recognition. The distinction between knowledge recognition and data mining is not very strict. It is called data mining in the application of technology. However, some scholars regard data mining and knowledge recognition as the same idea, and think that data mining pays more attention to process. At present, there are many researches on data mining. The following research results are analyzed as examples. In the context of large data, reference [9] proposed a classi-

*Corresponding author. E-mail: donglianga1@126.com.

fication mining algorithm S-CVFDT based on Storm. In the process, the parallelization window is combined with S-CVFDT algorithm to detect the mutation conceptual drift in the data stream center through the parallelization window, so as to realize the adaptive size change of the parallel window. At the same time, the concept drift model of progressive nature is continuously updated by S-CVFDT algorithm. Reference [18] proposes a data mining algorithm based on boundary marking window technology. In the process, the first window to be processed in the data stream is identified as the boundary marking window and processed accordingly. The maximal canonical patterns in each window can be obtained by the increment of the maximal set of canonical patterns in the previous window. To improve the efficiency of data stream mining, a hierarchical data mining algorithm based on Boolean reduction series is proposed in reference [12]. In the process, Boolean transform is applied to the data stream sequence based on the two sequence values of the original data stream, so as to reduce the Boolean reduction calculation cost. The number of elements in a sequence is reduced by using the transformation and reduction of sequence elements.

The efficiency and accuracy of the above data mining algorithms need to be improved. A distributed data flow mining algorithm based on matrix weighted association rules is proposed.

2. Distributed data stream mining based on matrix weighted association rules

The construction process of mining algorithm based on matrix weighted association rules is shown in Fig. 1.

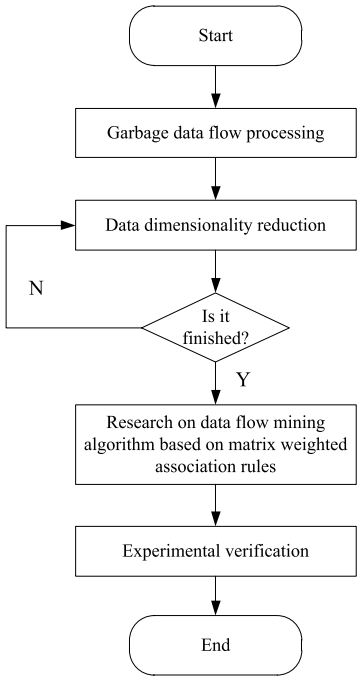


Fig. 1. Algorithm construction process.

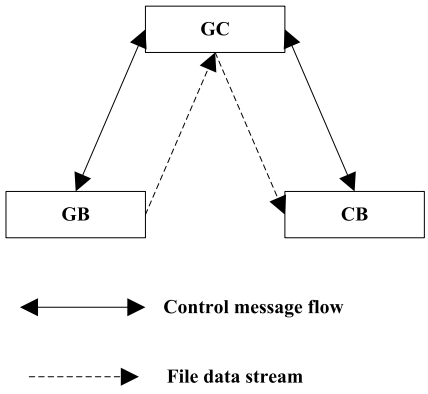


Fig. 2. Partition of garbage data processing module.

2.1. Garbage data stream processing

In this paper, according to the separation of metadata and data flow, CStore system saves distributed data flow in different clusters to realize garbage data processing. Based on garbage data processing, different tasks are performed on distributed data flow service nodes, which are divided into three parts: global control GC, bitmap generation GB, and data judgment CB, as shown in Fig. 2.

In the figure, the global control unit mainly distributes tasks for the other two units, and also provides the function of interface for managers to get the status information of current data flow; the bitmap generating unit regards the bucket as the unit, carries out bitmap identification operation for each data block symbol, and uploads bitmap files at the same time; and makes use of data judgment. The unit judges each data mark in each bucket based on the merged bitmap, transmits the result to the service unit, and realizes the garbage data processing.

Each valid data flow information save the GB unit on the MU node of the cluster and the CB unit in the data block information saves the CB unit in the SU of the cluster. All of them need a connection to communicate with the GC unit on the CS node of the man-

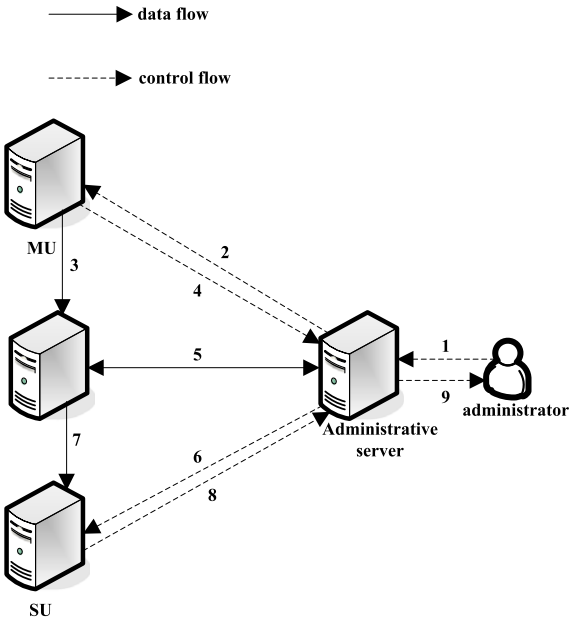


Fig. 3. Execution process of garbage data processing.

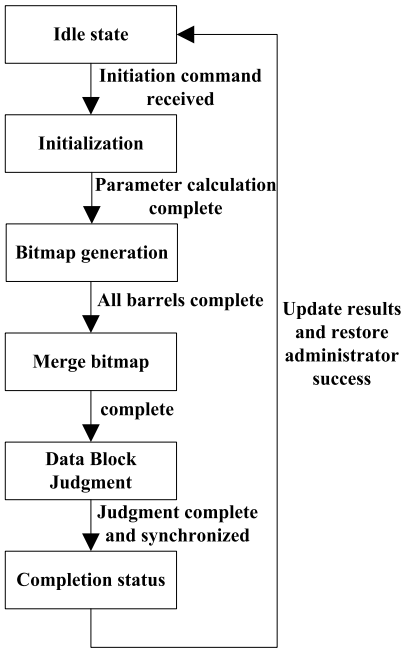


Fig. 4. State change and trigger state change of GC cell.

agement server. The connection sharer is the heartbeat information and task data, so that the heartbeat information can accurately judge whether the connection is normal or not. With different connections, normal heartbeat connections may occur, but when there is an abnormal connection between real business data, the heartbeat information cannot detect the abnormality. Figure 3 shows the execution process of garbage data processing.

According to the above structure diagram design, the garbage data stream processing is completed from the following units.

In the implementation of global control unit, GC unit is mainly responsible for global control in garbage data processing. It first monitors the commands of global garbage data processing transmitted by administrators, and then transmits detailed garbage processing commands to all CB nodes one by one. It is also responsible for merging and flow control of bitmap files transmitted by all GB units [14,19,20].

GC unit state machine: GC unit is mainly responsible for the control process, continuous transition between states. Figure 4 shows the state change and trigger state change of the unit.

In Fig. 4, when garbage data processing is not started, the whole processing program is idle. After starting, GC unit calculates initialization data, notifies GB to generate bitmap, realizes bucket selection and task publishing based on current rules, and actively transmits heartbeat connection after receiving garbage data processing command. Each time the GC completes the generation of a barrel bitmap, uploads it and notifies the GC whether the detection is complete or not. Suppose that all bitmaps with the same location are implemented and operated so as to obtain a bitmap file and realize the identification of valid data block bitmaps. When the above operation is completed, the CB unit is started and downloaded and merged. Bitmap, and then verify the address of the data blocks in each bucket. After verification, the results are transmitted to GC nodes, and the administrator replies that the garbage data processing is completed.

The model of bitmap generation and data judgment implementation is shown in Fig. 5.

According to the above model, GB consists of three threads, the main thread is the communication thread, and two sub-threads are created during startup. The main responsibility of the business thread is to execute the business logic, that is, to generate bitmap files, while the status thread is only responsible for obtaining the current load.

The bitmap generation business process is shown in Fig. 6.

When the GB node receives the command to start the garbage data processing from the GC transmission, it needs to perform the following operations: retain all the parameters of the garbage data processing,

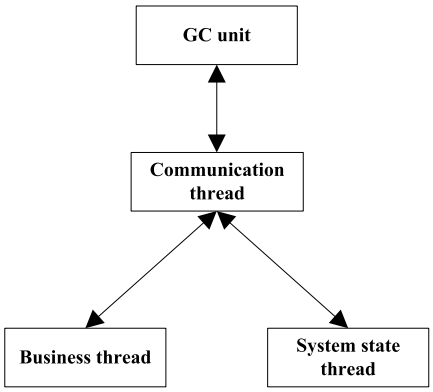


Fig. 5. Bitmap generation and data judgment implementation model.

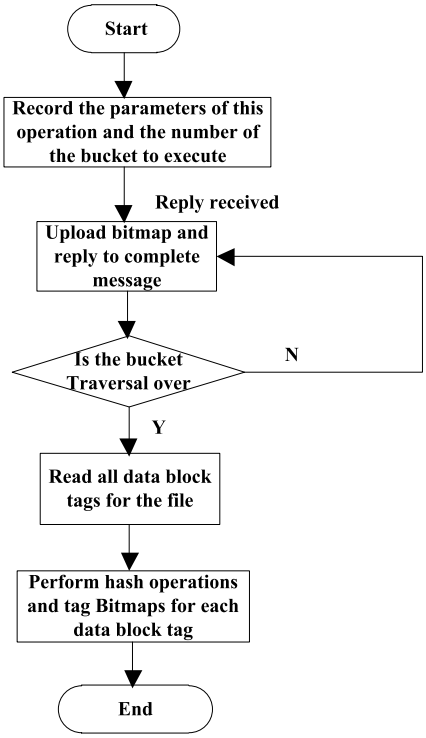


Fig. 6. Bitmap generation business process.

configure the memory based on the specified bitmap size in the parameters, decompose the memory into blocks according to the number of bitmaps, and then the main thread replies the information that the GC node is ready to wait. At a random interval, the heartbeat message is transmitted to the main thread. At the same time, the heartbeat message is transmitted regularly before the garbage data processing. When the GC node receives the heartbeat message for the first time, it is required to issue tasks to the GC node.

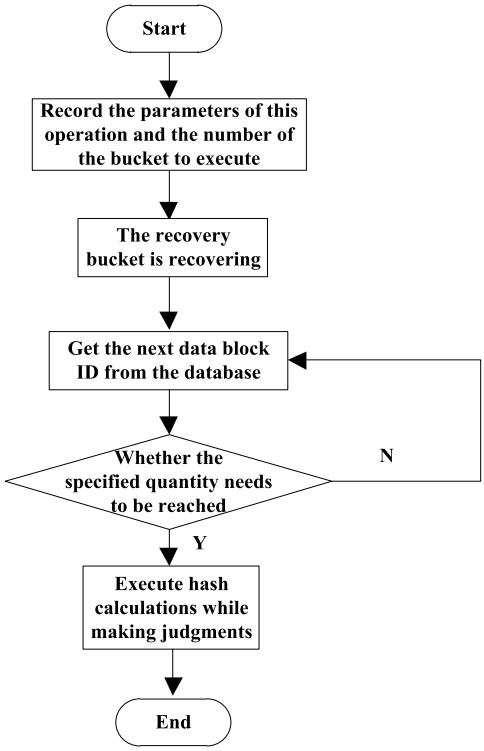


Fig. 7. Data judgment unit business process.

The business process of the data judgment unit is shown in Fig. 7.

When CB node receives the command of starting garbage data processing transmitted from GC node, the operation to be performed is to retain all the parameters of the garbage data processing, read the bitmap file in batches based on the parameter information, read the specified data block beforehand, and then the main thread replies to the information that GC node is ready for a period of time. At random intervals, heartbeat messages are transmitted to the main thread. At the same time, heartbeat messages are transmitted regularly before the garbage data processing. When a GC node receives a heartbeat message for the first time, it will issue tasks to the node.

2.2. *Data dimension reduction*

Based on the above garbage data processing results, the sliding window and data outline structure are used to optimize the principal component analysis algorithm and reduce the dimension of the data. Figure 8 is a sliding window schematic diagram:

According to the above figure, we can see the details of sliding window implementation. At this time,

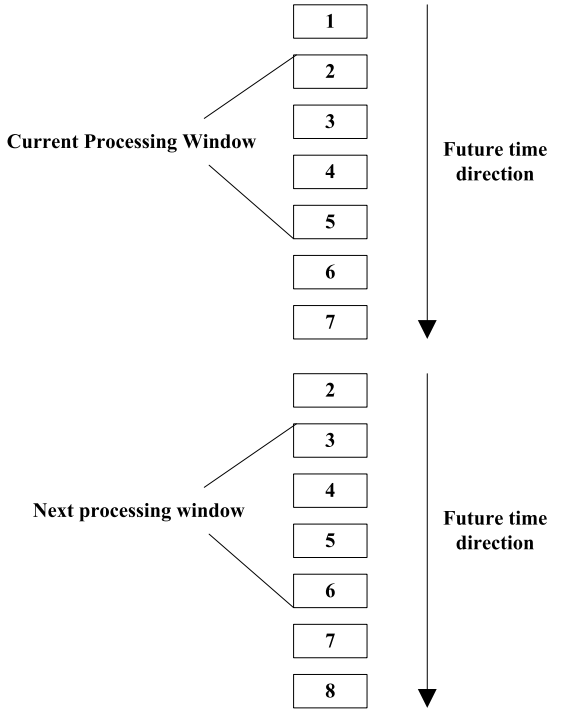


Fig. 8. Schematic diagram of sliding window.

the saved items in sliding window are 2, 3, 4, 5. When the new data items reach, the two at the tail will be discarded, the new data items will be added to the window, and the window will continue to slide forward along the continuous inflow of new data.

The data dimension reduction algorithm is divided into two parts, one is the first window principal component analysis, the other is the incremental principal component analysis based on the results of the first part [5,10,16]. The detailed process is shown in Fig. 9.

According to Fig. 9, there are:

Step 1: Identify the data attributes in the input data stream. Assuming that the attributes only contain numerical attributes, then go directly to step 2. Conversely, process the nominal attributes, convert them into numerical attributes, and complete the unification of data attributes.

Step 2: Set a time label for each data stream, regard the label as the size of sliding window, set the granularity of 1s as the window, assume that the time label of data is the same, then these data will be processed in the same window, and the next 1s data will be the next window.

Step 3: Calculate the correlation coefficient matrix for the generated outline data structure, then calculate the eigenvalues and eigenvectors for the eigenvalues, calculate the contribution rate for the eigenvalues, select the principal component based on the contribution rate, and then select the eigenvectors corresponding to each principal component to judge the principal component decision matrix [1–3].

Step 4: The principal component decision matrix is regarded as the whole sliding window decision matrix. All data in the window are mapped to the decision matrix subspace to complete the data dimension reduction operation.

Step 5: Slide the sliding window along the front of a basic window, then the window processing is different from the first one. Select the first n data to form a summary data set, calculate the correlation coefficient matrix R_i, and then project R_i into space according to the unit matrix H_{i-1} of the correlation coefficient matrix of the previous window. In summary, there are:

$$H_{i-1} = \frac{1}{n} X_{i-1} V_{i-1} A_{i-1}^{-1}. \tag{1}$$

In the formula, X_{i-1} represents the outline data set, V_{i-1} represents the selected k principal component, A_{i-1}^{-1} represents the selected first k eigenvalues.

Based on the above calculation and analysis, the projection R_i in H_{i-1} tensor space can be expressed as:

$$\overline{R_i} = H_{i-1}^{T} R_i H_{i-1}. \tag{2}$$

The eigenvalues and eigenvectors are calculated for $\overline{R_i}$ and arranged in descending order. K eigenvalues A_i and their corresponding eigenvectors V_i are selected. Based on these K features, incremental eigenvalues $A = \frac{1}{2n}(I + A_i)$ and eigenvectors $V = H_{i-1} V_i$ are obtained.

Based on the above calculation, the data in the window form a decision matrix according to the obtained eigenvectors to realize dimensionality reduction mapping, and then the window continues to iterate the algorithm operation of the window.

Summarize the above process, unify the data attributes, fix the data needed to reduce dimension according to the sliding window in the data stream with unified numerical attributes, and then construct an outline structure in the window to get the principal component decision matrix, which is used to realize the data dimension reduction in the sliding window. It not only satisfies the real-time processing of data stream, but also provides support for subsequent data mining.

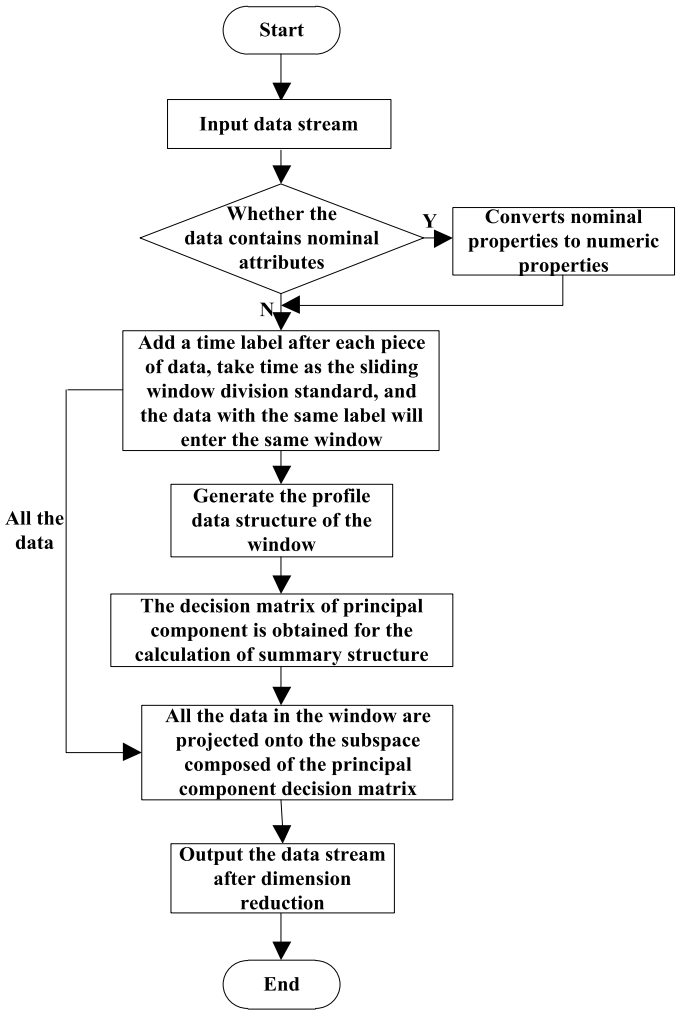


Fig. 9. Data dimension reduction.

2.3. *Data stream mining based on matrix weighted association rules*

Matrix weighted association rules have the characteristics of high comprehensiveness and high efficiency in data mining. Using matrix weighted association rules can meet the design requirements of data flow depth mining algorithm. When mining the distributed data algorithm with matrix weighted association rules, first cluster the transactions in the database according to the time distribution, then carry out weighted analysis on each aggregation, and output the weighted frequent item set with time.

Distributed data stream preprocessing:

(1) Specify the number of clusters that need to be distinguished k^* and the time threshold E.

(2) Give the distance between two transactions $d = s\ (s = |s_1 - s_2|)$, where s_1 and s_2 represent the generation time of two records respectively.

(3) Arbitrarily select k^* transactions as the initial centroid in the database.

(4) Based on the distance between the transactions and the initial centroid, the transactions are divided into data clusters nearest to the initial centroid in a certain order [8,13,21].

(5) The centroid of each data cluster is recalculated until the centroid no longer changes.

(6) Based on the above steps, transaction databases can be divided into k^* smaller databases D_{k^*}.

(7) Give the corresponding weight to each item in k^* databases. Assuming that transaction databases

are arranged in chronological order $\{D_1, \ldots, D_{k^*}\}$, the weight values of each item in the jth database can be expressed as follows:

$$W_i = \frac{1}{(k^* - j + 1)} \frac{P_i}{P_{max}}. \quad (3)$$

(8) Scanning the database D_{k^*} after clustering, and constructing Boolean matrix A^*, so that each column in the matrix can store one or more transaction information to ensure that there are no duplicate columns.

According to the preprocessing operation, the distributed data stream mining algorithm based on matrix weighted association rules is as follows:

The input is transaction database D_{k^*}, weight set W_{k^*}, weighted support minimum minsup, and confidence minimum minconf.

The output is weighted frequent itemsets in D_{k^*}.

The detailed process is as follows:

(1) Obtaining weighted frequent itemsets.

The number of occurrences of projects in all transactions is calculated.

$$SC(\{i_j\}) = \sum_{k^*=1}^{n} (a_{jk^*} \times C[k^*]). \quad (4)$$

Assuming that the support degree of a certain item set is greater than or equal to the minimum support degree required for it to become a weighted frequent item set, that is, $SC(\{i_j\}) \geqslant SC_{min}(\{i_j\})$, the corresponding row vectors of this set are marked. At this time, the itemset corresponding to the marked row vectors is the weighted frequent itemset.

Assuming that the support degree of an item set is smaller than the minimum support expectation value B_{min}, the row vectors corresponding to the item set will be deleted and saved instead. A^{*1} is obtained by combining the preserved data according to the original order.

For the sum of columns in A^{*1}, if the result is smaller than 2, then delete the column vector.

For the saved column vectors in A^{*1}, they are combined according to their original order, and A^{*2} is obtained.

(2) Obtaining weighted frequent itemsets.

For each line in A^{*2}, the operations are carried out in bits. At the same time, the number of times the two items appear together in all the transactions is calculated according to the results obtained.

$$SC(\{i_q, i_j\}) = A_q^* \wedge A_j^* = \sum_{k^*=1}^{n} ((a_{qk^*} \wedge a_{jk^*}) \times C[k^*]). \quad (5)$$

Assuming that the support degree of a set of two items is greater than or equal to the minimum support degree required for it to become a weighted frequent item set, that is, $SC(\{i_q, i_j\}) \geqslant SC_{min}(\{i_q, i_j\})$, then the row vectors corresponding to the set of items are marked, and then the set of items corresponding to the marked row vectors is the weighted frequent item set.

Assuming that the support degree of a set of two items is smaller than the minimum support expectation value B_{min}, the row vectors corresponding to the set will be deleted and saved instead. The preserved row vectors are reassembled according to the original order, and the matrix A^{*3} is obtained.

For each column in A^{*3}, if the result is less than 2, then delete the column vector.

For column vectors preserved by A^{*3}, they are combined according to their original order, and A^{*4} is obtained.

(3) Obtaining weighted frequent k^* itemsets.

The rows of A^{*4} are processed bitwise, and 1 of the results is filtered out, multiplied and added with the corresponding column weight values to obtain the support degree of the set. Assuming that the value is greater than or equal to the minimum support degree required to become a weighted frequent itemset, then the row vector of the set is marked, and then the support degree of the set is obtained. The corresponding itemsets of the marked row vectors are weighted frequent three or four itemsets.

Assuming that the support degree of an itemset is smaller than the minimum support expectation value B_{min}, the corresponding row vectors of the itemset will be deleted and saved instead. The saved duplicate row vectors are deleted and reassembled according to the original order, and A^{*5} is obtained.

For each column in A^{*5}, if the result is less than 2, then delete the column vector.

For column vectors preserved by A^{*5}, the original sequence is recombined to obtain A^{*6}.

Iterating the above process until the number of rows of the matrix is less than or equal to 1, the algorithm terminates.

(4) The row vectors completed by all tags are transformed into corresponding weighted frequent itemsets, and the mining results are output at the same time.

3. Experiments and discussions

In order to verify the effectiveness of the distributed data stream mining algorithm based on matrix weighted association rules, an experiment was conducted. The experimental environment is shown in Table 1.

The experimental indicators are as follows:

(1) Mining efficiency (based on the time to get frequent itemsets).

(2) Accuracy of mining.

The specific experimental scheme is: using this algorithm and reference [9], reference [18] and reference [12] algorithm to compare the above experimental indicators. The experimental data is the FoodMart database contained in mysql, with a data size of 150 GB.

Table 1
Experimental environment

Hardware	Software
CPU Intel Core 2 Duo T8100 2.1 GHz	Microsoft Windows 7 operating system
2 GB memory	Java programming
320 G hard disk	

3.1. Comparison of mining efficiency

According to Fig. 10, when the data volume is 10000 bit, the mining time of the four algorithms is 20 s. With the increasing data volume, the mining time of the four algorithms is on the rise. When the data volume reaches 90000 bit, the mining time of the reference [9] algorithm is 200 s, the mining time of the reference [18] algorithm is 199 s, and the mining time of the reference [12] algorithm is 194 s. The mining time of the algorithm is 162 s, which is far less than the three literature algorithms, which fully shows that the method in this paper has high mining efficiency. In the process of constructing the algorithm in this paper, cstore system is used to store the distributed data flow in different clusters according to the way of separating metadata and data flow, so as to realize garbage data processing, which greatly improves the mining efficiency of the algorithm.

3.2. Comparisons of mining accuracy

In order to further verify the data stream mining performance of the proposed algorithm, experimental verification is carried out with mining accuracy as a comparison index. The results of comparison of mining accuracy of the four algorithms are shown in Fig. 11.

As can be seen from Fig. 11, the data mining accuracy of the algorithm proposed in this paper is higher than that of the literature algorithm, showing good ap-

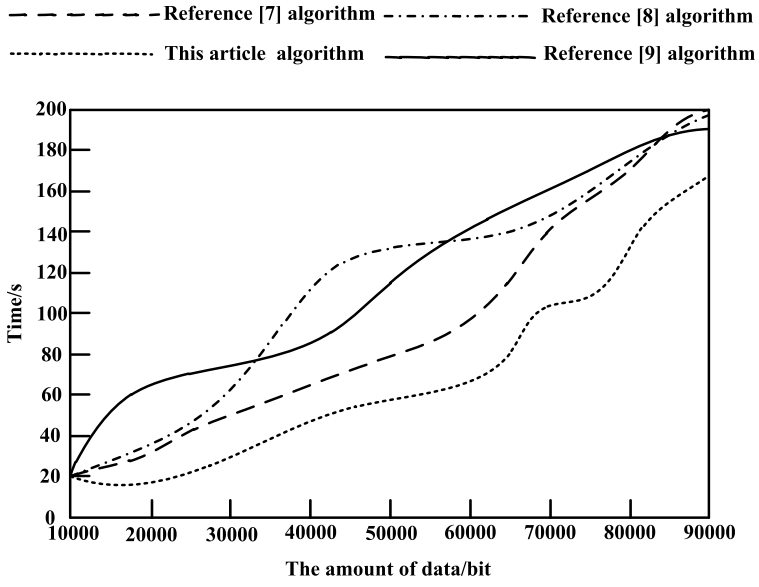
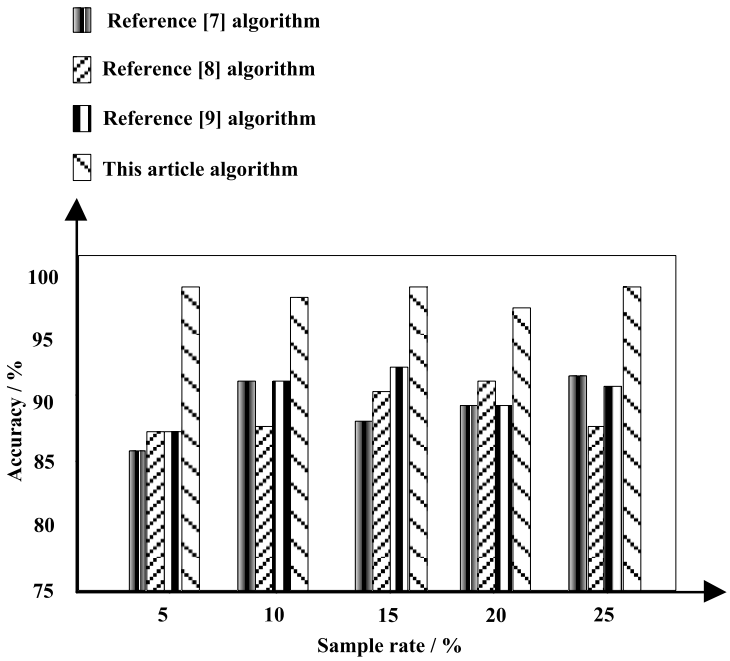
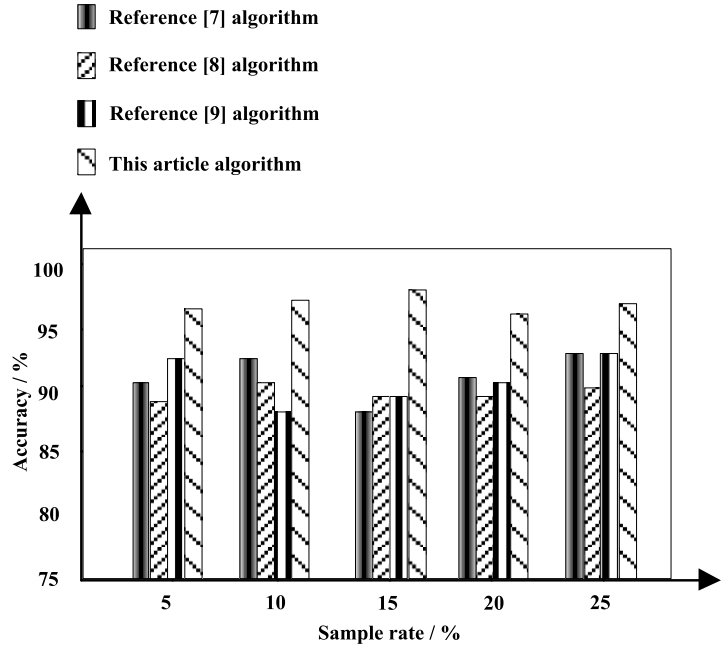


Fig. 10. Comparison of mining efficiency of different algorithms.

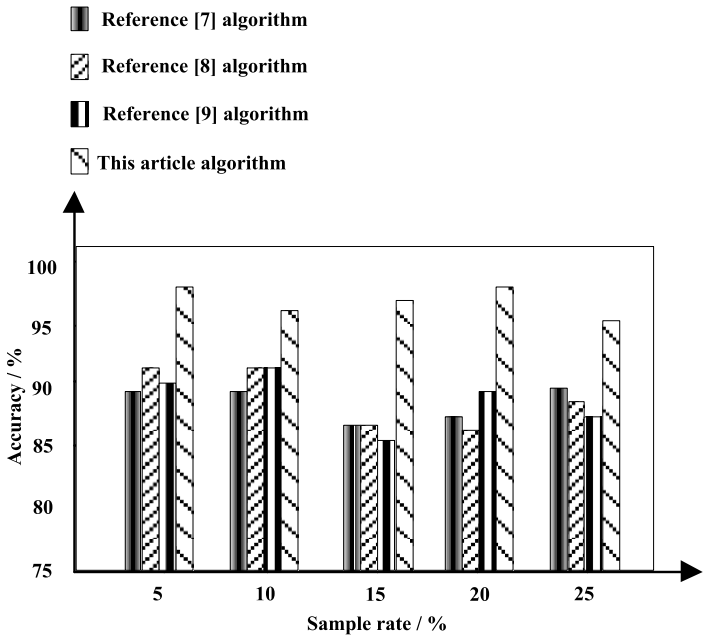


(a) Accuracy of different algorithms under 10 tests



(b) Accuracy of different algorithms under 20 tests

Fig. 11. Comparison of mining accuracy of different algorithms.



(c) Accuracy of different algorithms under 30 tests

Fig. 11. (Continued.)

plication performance. When the algorithm uses matrix weighted association rules to mine the distributed data algorithm, it first clusters the transactions in the database according to the time distribution, then carries out weighted analysis on each aggregation, and outputs the weighted frequent term set with time. It improves the accuracy of data mining based on matrix weighted association rules.

4. Conclusions

In the context of big data, with the increasing number of distributed data streams, the effectiveness of traditional data stream mining algorithms needs to be further improved. Therefore, this paper proposes a distributed data flow mining algorithm based on matrix weighted association rules. The following conclusions are proved in theory and experiment. This algorithm has good efficiency and accuracy in mining large-scale data stream. Specifically, compared with the storm based mining algorithm, the mining efficiency is greatly improved, and the maximum mining time is only 162 s; compared with the boundary marker window based mining algorithm, the mining accuracy is significantly improved, and the maximum mining accuracy is 98.9%. Therefore, the mining algorithm based on matrix weighted association rules proposed in this paper can better meet the requirements of distributed data flow mining. In the future work, we should further improve the accuracy of mining to meet the needs of big data growth.

References

[1] D. Chao, J. Lin, W. Fang et al., Data-driven affinely adjustable distributionally robust unit commitment, *IEEE Transactions on Power Systems* **33**(2) (2018), 1385–1398. doi:10.1109/TPWRS.2017.2741506.

[2] F.M. Chen, D.Z. Han, K. Bi et al., Key technologies of distributed data stream processing based on big data, *Journal of Computer Applications* **37**(3) (2017), 620–627. doi:10.3724/SP.J.1087.2012.00620.

[3] H. Dan, D. Han, J. Wang et al., Achieving load balance for parallel data access on distributed file systems, *IEEE Transactions on Computers* **67**(3) (2018), 388–402. doi:10.1109/TC.2017.2749229.

[4] G. Ercolani and F. Castelli, Variational assimilation of streamflow data in distributed flood forecasting, *Water Resources Research* **53**(1) (2017), 158–183. doi:10.1002/2016WR019208.

[5] Z. Jiang, Z. Zhang and G. Hui, Towards secure data distribution systems in mobile cloud computing, *IEEE Transactions on Mobile Computing* **16**(11) (2017), 3222–3235. doi:10.1109/TMC.2017.2687931.

[6] K.W. Lin, S.H. Chung and C.C. Lin, A fast and distributed algorithm for mining frequent patterns in congested networks,

Computing **98**(3) (2016), 235–256. doi:10.1007/s00607-015-0457-6.

[7] Q.L. Lin and S.Z. Yu, A distributed green networking approach for data center networks, *IEEE Communications Letters* **21**(4) (2017), 797–800. doi:10.1109/LCOMM.2016.2642188.

[8] Y. Lin, Research on data mining of valuable information in unstructured network, *Computer Simulation* **34**(2) (2017), 414–417.

[9] L.L. Lu, Y.P. Zhang, H.Y. Tan et al., Research on classification algorithm and concept drift based on big data, *Journal of Frontiers of Computer Science & Technology* **10**(12) (2016), 1683–1692.

[10] M. Misaki, T. Tsuda, S. Inoue et al., Distributed database and application architecture for big data solutions, *IEEE Transactions on Semiconductor Manufacturing* **30**(4) (2017), 328–332.

[11] D.G. Murray, P. Barham, P. Barham et al., Incremental, iterative data processing with timely dataflow, *Communications of the ACM* **59**(10) (2016), 75–83. doi:10.1145/2983551.

[12] Y.G. Ren, H.Z. Qian, Lang and H.Y. Lag, Correlation mining method based on improved Boolean reduction and layered series for big data stream, *Pattern Recognition and Artificial Intelligence* **29**(5) (2016), 455–463.

[13] B. Ru and X.Z. He, High utility itemsets mining algorithm of data stream with reducing candidate itemsets, *Application Research of Computers* **34**(11) (2017), 3379–3383.

[14] F. Tang, H. Zhang, L. Fu et al., Distributed stable routing with adaptive power control for multi-flow and multi-hop mobile cognitive networks, *IEEE Transactions on Mobile Computing* **18**(2) (2019), 2829–2841. doi:10.1109/TMC.2018.2885762.

[15] E. Tomes, E.N. Rush and N. Altiparmak, Towards adaptive parallel storage systems, *IEEE Transactions on Computers* **67**(12) (2018), 1840–1848. doi:10.1109/TC.2018.2836426.

[16] R. Tripathi, S. Vignesh and V. Tamarapalli, Optimizing green energy, cost, and availability in distributed data centers, *IEEE Communications Letters* **21**(3) (2017), 500–503. doi:10.1109/LCOMM.2016.2631466.

[17] S. Wang, L. Huang, Y. Nie et al., Local differential private data aggregation for discrete distribution estimation, *IEEE Transactions on Parallel and Distributed Systems* **30**(9) (2019), 2046–2059. doi:10.1109/TPDS.2019.2899097.

[18] Wen, Y.Y. Wen, S.P. Wang and H. Zhao, The maximal regular patterns mining algorithm based on landmark window over data stream, *Journal of Computer Research and Development* **54**(1) (2017), 94–110.

[19] H. Zhang, B. Zhang, A. Bose et al., A distributed multi-control-center dynamic power flow algorithm based on asynchronous iteration scheme, *IEEE Transactions on Power Systems* **33**(2) (2018), 1716–1724.

[20] T. Zhang, R. Shu, Z. Shan et al., Distributed bottleneck-aware coflow scheduling in data centers, *IEEE Transactions on Parallel and Distributed Systems* **30**(7) (2019), 1565–1579. doi:10.1109/TPDS.2018.2889685.

[21] Y. Zhang, Y.K. Bao, L.S. Shao et al., A multivariate decision tree for big data classification of distributed data streams, *Acta Automatica Sinica* **44**(6) (2018), 157–169.