EAI Endorsed Transactions
On Scalable Information Systems
Research Article EAI.EU

Database System Based on 3Dmax Big Data Mining Technology

Xiaoyu Chen[1] Junkai Zhang[1,*] Pengshan Ren[2] Xian Hua[1] Yanfeng Ni[1]

[1] Engineering Technology Education Center, Henan Institute of Technology, Xinxiang, 453003, Henan, China
[2] School of Electronic Information Engineering, Henan Institute of Technology, Xinxiang, 453003, Henan, China

Abstract

INTRODUCTION: When performing data retrieval, various databases have high real-time performance. The general mining method is prone to data failure, resulting in real-time data loss, which requires using association rule mining algorithms for large data sets to solve.
OBJECTIVES: This project intends to study the mining method of FP-growth frequent entries in 3Dmax big data under the framework of Hadoop, combined with the Map Reduce development model.
METHODS: First, select the transaction database according to each frequency and generate the corresponding projection database. Then the obtained image database is distributed on each node computer. Furthermore, under the guidance of the node machine, the projection is divided into different regions to generate several smaller sub-databases. The method was mined in parallel using node machines to generate local frequency items. Finally, all local frequency sets are merged into a complete frequency set.
RESULTS: For the database segmentation algorithm, the larger the minimum support value is, the less frequently a 1-item projection database will be generated. The less time it takes to distribute data. It also spends less time mining the projection subdatabase. Compared with the FP-growth algorithm, the speedup ratio of the database segmentation algorithm does not increase linearly with the increase of the minimum support value.
CONCLUSION: This method does not need to generate many FP trees like the conventional FP-Growth method. This method can better overcome the problem of calculation failure caused by the limited memory of a single computer in the conventional FP-Growth method and several other methods. At the same time, because the sub-libraries of the partitions are similar in size, the load allocated to each node machine is more balanced. This improves the effectiveness of the algorithm.

Keywords: Frequent item set; 3Dmax; FP-Growth; Big data mining; Map Reduce Programming Model

Received on 16 August 2023, accepted on 19 September 2023, published on 19 September 2023

Copyright © 2023 X. Chen *et al.*, licensed to EAI. This is an open access article distributed under the terms of the CC BY-NC-SA 4.0, which permits copying, redistributing, remixing, transformation, and building upon the material in any medium so long as the original work is properly cited.

doi: 10.4108/eetsis.3727

*Corresponding author. Email: jk09060421@163.com

1. Introduction

Data is also known as knowledge mining in databases. It refers to discovering unknown and potentially valuable information from the data of a transactional database. It involves e-commerce, government management, engineering applications, commercial applications, etc. Massive data, continuous growth, diverse attributes, etc, characterize it. According to its application scope, data mining can be divided into many types: association rule mining, classification, sequence pattern mining, cluster analysis, etc. Association rules are studied from the perspective of goods matching in a shopping basket. The main goal of this method is to obtain the interrelation between the different attributes in the transaction set to obtain reliable and highly correlated rules. The system will use those relevant rules to assist. These problems include placing and collocating commodities in

supermarkets, curriculum settings in colleges and universities, and consumer purchasing habits in e-commerce. The requirement association rule mining based on these applications has attracted many scholars' research and exploration.

Agrawal first proposed the AIS algorithm and developed it into two main association rule mining algorithms. One is Agrawal as an Apriori-based method for generating frequent candidate sets. The second is an FP-based algorithm that does not produce an FP tree of candidate sets. Both Apriori and FP-Growth methods have their advantages and disadvantages. Literature [1] and literature [2] are improved algorithms for frequent item set mining using different parallel processing techniques based on the FP-Growth algorithm. They increase the efficiency of the algorithm. But they face the same problems. In the big data environment, the generated FP tree cannot be stored in a single machine when the large-scale transaction database is used, which results in algorithm failure. Literature [3] uses mapping database technology and the programming pattern of Map Reduce and parallel processing technology to achieve this. The above problems have been satisfactorily solved. However, in the specific operation process, there are often problems such as limited resources provided by nodes and insufficient storage space of a single node. This limits the use of this method. In some extreme cases, a projected database can be as large or even close to the size of the original transaction database and thus also cause the algorithm to fail. This paper introduces a method of data segmentation. The user can set the size of the segmented sublibrary to effectively avoid the problem of unavailability caused by a single memory resource in reality.

2. Mining of dynamic database association relationships under large data sets

Compared with a static database, the data in a dynamic database is divided into two types: historical data and new data. Therefore, static and dynamic mining are combined in this paper when selecting the dynamic database association data mining method. The first step is to obtain information about the historical data from the dynamic database [4]. Then the new data in the dynamic database is mined to ensure the correctness of data mining.

2.1 Dynamic database data storage

First, according to specific laws, historical big data is divided into multiple subcategories. Many small files need to be integrated into a sub-data set to improve the storage efficiency of large amounts of data. At the same time, statistical techniques are applied to each sub-data set to improve the quality of massive data. Label data space and time as O. The node uses the letter D to represent the data on the node. In the distributed data storage, the respective small files and sub-datasets are put into the corresponding nodes according to the partition mode. A new computational model is proposed to accurately measure the support degree of each node in the heterogeneous network [5]. Then the data that does not meet the requirements for support is removed. Finally, the sequential data is obtained, and the target of distributed data storage is achieved.

2.2 Mining dynamic databases

After the scientific grouping of the data in the dynamic database, the packet data information before the dynamic database can be obtained at the node [6]. After analyzing and integrating it, it can get < key= word, value= {g1, g2, ... gN}>. Consider this set of data as input data in data mining. In this way, a schematic diagram of the data mining process of existing dynamic databases can be produced (Figure 1 quoted in Ecological Modelling. 295.5-17).

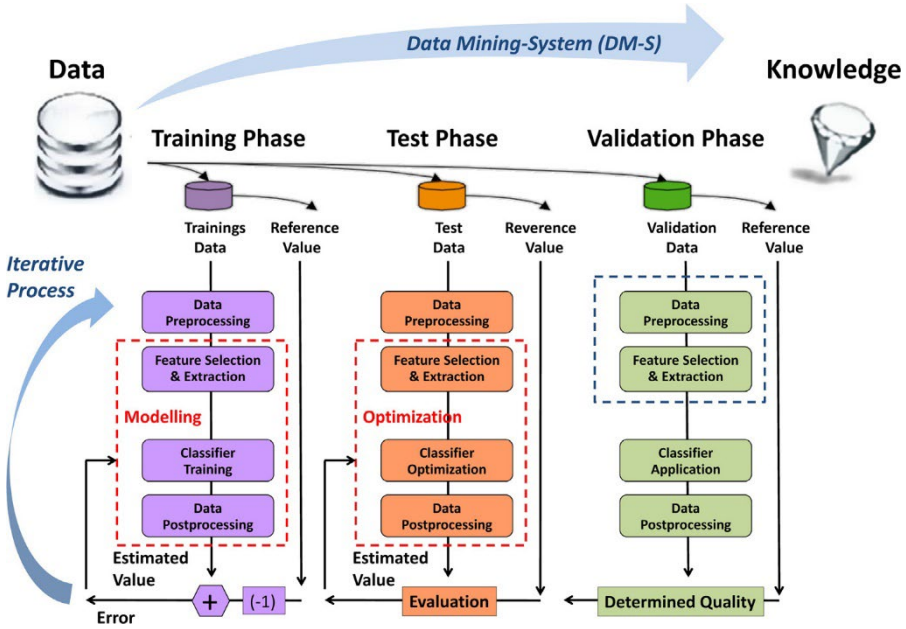

Figure 1. The existing dynamic database data mining process diagram

In the schematic of this process, the key enters new data into a dynamic database. When mining the existing data of a dynamic database, it is necessary to combine different types of data [7]. Then enter in the specified way. Whenever new data is entered into the database, a corresponding change occurs. This creates a persistent data tree. Merges of frequent collections are not recalculated until after the data is output and the data tree is restored to its original state, completely updating the existing data information. The dynamic database automatically updates its data when new data is entered. However, due to the difference in the input data type, different new data scenarios will appear in the actual update process. According to different situations, the dynamic database must adopt corresponding mining methods. Figure 2 shows a newly added data mining process (the image is cited in a high-speed railway network dataset from train operation records and weather data).

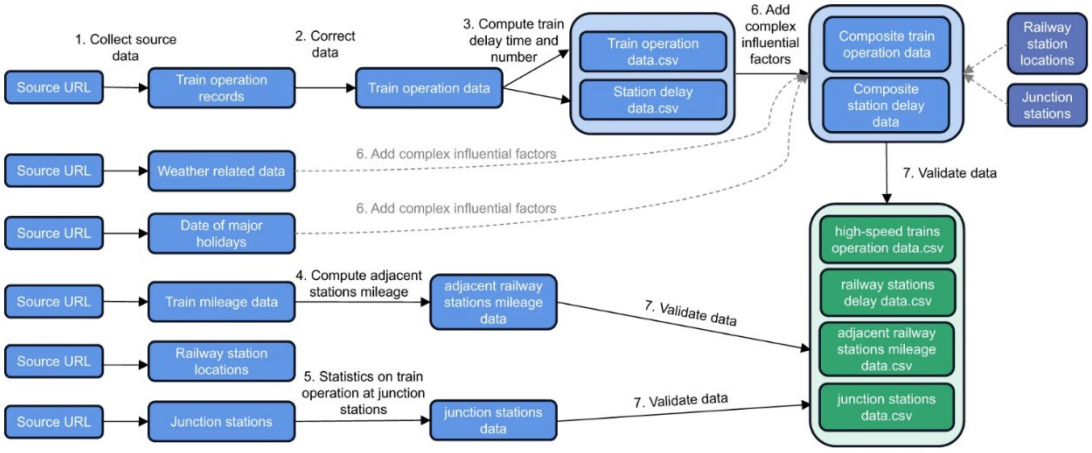

Figure 2. Dynamic database diagram of the new data acquisition process

For case two, it needs to calculate how often it happens. The frequency of the data can be obtained by superimposing the newly added frequency item set with the original frequency item set. The item set in Scenario 3 recalculates the data for all nodes to obtain the frequency of the item set in the overall data [8]. The entries in case 4 always appear infrequently in dynamic databases. There is no more information to be mined. According to the three

cases of the above plan, the newly added data information of the dynamic database is effectively mined. It is possible to combine existing and newly added data. This allows you to access relevant information in a dynamic database.

3. Classic FP-Growth approach

A data structure called FP-Tree is used in the FP-Growth algorithm. FP tree is a kind of tree structure with a specific semantic structure. It consists of a frequent item table and an item prefix tree. The method is divided into two steps. The first step is compressing all the transaction data into a frequent pattern tree. The second step is to get the complete set of frequency items from the frequency map.

In this paper, an example is given to illustrate the implementation steps of the method [9]. Assume that the transaction database looks like the table in Table 1. This trade database contains ten trades. It includes projects $\alpha,\beta,\gamma,\delta,\varepsilon,\zeta,\eta,\theta$. Set the minimum support count to "Minimum Support" =2.

Table 1. Transaction database D

SID	Transaction ID list
S1	α, β, ε
S2	β, δ
S3	β, γ
S4	α, β, δ
S5	α, δ, ζ
S6	β, γ, δ
S7	α, γ, δ
S8	α, β, γ, ε
S9	α, β, γ, η
S10	α, β, θ

The main steps of the algorithm are as follows:

- The first scan of transaction database D to obtain all frequent projects. And the frequent 1 item is sorted in descending order of support count to obtain the frequent 1 item header table R (Table 2). The number of ξ,η and θ support degrees is 1. It is below the minimum support number and is an infrequent item, so they are not in the frequent 1 item set table R.

Table 2. Frequent 1-item header table R

Item	Support count
β	8
α	7
γ	6
δ	4
ε	2

- FP tree construction: establish a root node with a "null" flag. The database was checked a second time. For each transaction, a branch is created in the FP tree. Each node in the branch corresponds to each entry of the transaction [10]. And arrange them in the order shown in the table. Each element in the branch is increased by one. The FP tree structure is obtained by combining the frequent 1-bar table with the FP tree structure (Figure 3).

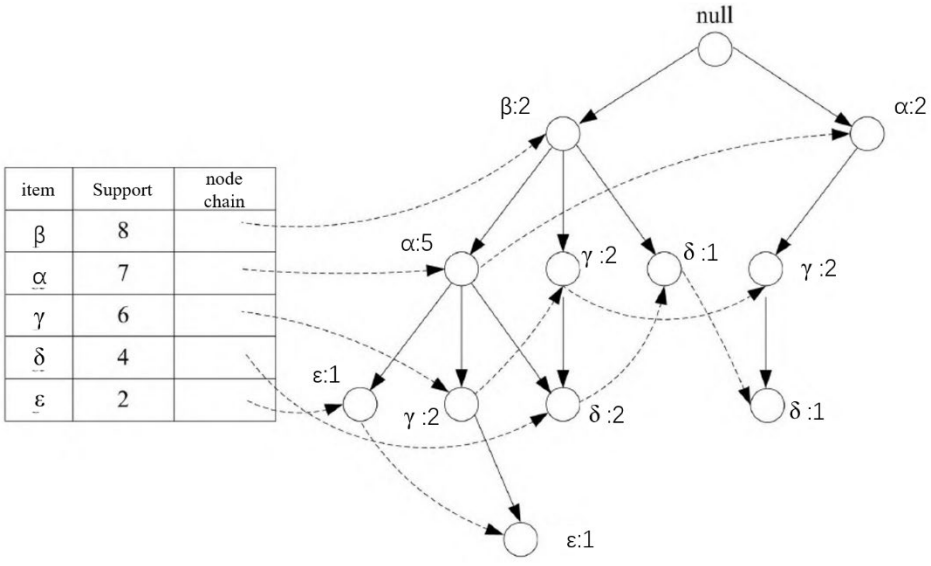

Figure 3. FP tree

- All frequent item sets are obtained by mining the above-generated FP tree;

4. Modify the FP-Growth algorithm

The calculation method of FP-Growth in transactional databases is not very effective. Fp-trees can process data in a single memory. But in big data, FP trees cannot be stored in a single memory. So that's not going to work. A new FP-Growth algorithm is proposed to achieve this goal. The papers still use the previous trade database D as an example to explain how specific improvements can be made [11]. The system used by this method is a parallel system, which includes a manager and multiple agent nodes. To implement the improved algorithm in such an operating environment, follow these steps:

- The transaction database D is scanned for the first time to obtain all frequent projects. And the frequent 1 item is sorted in descending order of support count to obtain the frequent 1 item header table R (Table 2). This step is the same as the regular FP-Growth algorithm.
- Perform data clearing operations of transaction database D. Remove an infrequently used item from D. Then, each time in D is extracted. For each "1" item, create a projection library containing that item. The projection database corresponding to $\alpha, \gamma, \delta, \varepsilon$ is shown in Tables 3 through 6.

Table 3. Projection database Dα corresponding to α

SID	Transaction ID list
S1	α, β, ε
S4	α, β, δ
S5	α, δ, ζ
S7	α, γ, δ
S8	α, β, γ, ε
S9	α, β, γ, η
S10	α, β, θ

Table 4. Projection database Dγ corresponding to γ

SID	Transaction ID list
S3	β, γ
S5	α, δ, ζ
S6	β, γ, δ
S7	α, γ, δ
S8	α, β, γ, ε
S9	α, β, γ, η

Table 5. Projection database Dδ corresponding to δ

SID	Transaction ID list
S2	β, δ
S4	α, β, δ
S6	β, γ, δ
S7	α, γ, δ

Table 6. ε corresponds to the projection database Dε

SID	Transaction ID list
S1	α, β, ε
S8	α, β, γ, ε

- There is still a high probability of FP trees directly generated from the projection library. It cannot be stored in a single memory [12]. The paper breaks down the projection database above into more detail. The projection library is divided into a single projection library based on a preset maximum number of transactions included. In the above example, you set up a partition containing up to 4 transactions. Each of the α and γ projection databases is then divided into two projective shadow databases $D_{\alpha:1}, D_{\alpha:2}$ and $D_{\gamma:1}, D_{\gamma:2}$ (Tables 7 to 10). A database of projected shadows is assigned to each node machine.

Table 7. Shadow casting database Dα:1 corresponding to α

SID	Transaction ID list
S1	α, β, ε
S4	α, β, δ
S5	α, δ, ζ
S7	α, γ, δ

Table 8.α corresponds to the shadow casting database Dα:2

SID	Transaction ID list
S8	α, β, γ, ε
S9	α, β, γ, η
S10	α, β, θ

Table 9. Shadow casting database Dγ:1 corresponding to γ

SID	Transaction ID list
S3	β, γ
S5	α, δ, ζ
S6	β, γ, δ

Table 10. Shadow casting database Dγ:2 corresponding to γ

SID	Transaction ID list
S7	α, γ, δ
S8	α, β, γ, ε
S9	α, β, γ, η

- Each node machine scans the projection library to build the shadow-casting tree structure of the corresponding project. An FP-tree construction method based on FP-Growth is proposed. $D_{m:i}$ k node machine is set to process the projection shadow database $D_{m:i}$, corresponding to m of the frequency 1 item. When processing each transaction in $D_{m:i}$, the items in each transaction are classified according to the order in table R. Then remove m and the other items. The remaining elements in the FP tree to be built only produce branches. Here is the detailed calculation method:
- Create a root node for the FP subdirectory and mark it as "null."
- Traverse database $D_{m:i}$. For each transaction in $D_{m:i}$:

Arrange the items in the transaction in the order in R. Then remove m and the other items. Record the list of transaction items as $[q|Q]$. Here q is the element of the first entry. Q is the list of the remaining entries. Call insert tree $([q|Q],S)$. Insert tree $([q|Q],S)$as follows: If S has a child N resulting in n.I.tem-name = p. tem-name, then the count of N is increased by 1. Increase the support rate of the item in the header table by 1, or create a new Node. Set its number to 1- link to its parent T. Add a project to that [13]. Support counter set to 1. Inserting tree $([q|Q],S)$ is called recursively when P is not null. The FP-tree generated according to this improved algorithm is called the FP-subtree $S_{m:i}$ of the projection database, which corresponds to the frequent 1 item m.

- Each node machine generates a local frequent item set by mining the FP subtree structure constructed.
- Gather the local frequency sets generated by the projection database of the same frequency 1 item in the same node machine and merge to form a frequency set with the same frequency 1 item.
- Finally, summarize the frequent item sets corresponding to the frequent 1 item generated by all node machines to obtain all the frequent item sets.

5. A way to discover parallel frequent entries

The method is divided into two steps: one is to generate all the "1 item" sets and create the "1 item" header table R; We can use the program pattern of MapReduce for parallel execution [14]. In the second stage, frequent entries are discovered by multiple node machines in parallel. The final sum will get all frequent items.

5.1 Performing Step 1

- Divide the transactions in the transaction database into the same n data blocks on the main computer. n packets of data are then transmitted to an independent node machine.
- Perform parallel mapping process for each enslaved person clearing unit. Please find a local project and their approval ratings. The projects are then put together and sent to the controller server.
- The primary server performs the restore process. Summarize all the results sent from the enslaved person. Find all 1-item sets and their support rates. The frequent set of 1 project is then placed in descending order by the value of the support count. Finally, the home page table R (Table 2) is obtained. The method uses a mapping model for parallel computation. Compared with the conventional FP-growth method, the calculation time is significantly reduced.

5.2 Perform Step 2

- Each node machine first performs data clearing of the data block allocated by the previous master. These data blocks are then filtered for one frequency. Generate a local projection database. This projection database is a frequently used item. All local projection databases of an item with the same frequency are concentrated on the same node [15]. Create a projection database for all records of an item with this frequency of 1.
- The secondary node machine will be divided into each projection library. The maximum number of transaction records included is set in advance. Divide the cast database into a single cast shadow database.

- Each agent node machine generates the corresponding FP subtree of the projection shadow library. A local frequent item set is obtained by exploring the FP subtree.
- The local frequent items generated by all sub-databases corresponding to the same projection database are aggregated to the same node. Finally, the results are sent to the controller server.
- The host summarized all the frequent item sets.

6. Algorithm experiment analysis

Due to the vast FP tree structure generated when processing a large amount of transactional data, it is difficult for the conventional FP-growth method to be effectively stored in a single memory. This problem that causes the algorithm to fail has been confirmed in the literature [16]. The algorithm given in this paper solves this problem because of the way of using the classifier. Therefore, this experiment will focus on the performance of our proposed segmentation database algorithm in parallel operations. The experimental method is mainly to compare the running time of the partition database algorithm in the Hadoop cluster environment with that of the traditional FP-Growth algorithm in the stand-alone environment. The Hadoop cluster system uses a master-slave architecture.

6.1 Test 1

A partitioned database algorithm is used in a clustering system consisting of 1 controller server +5 enslaved person binding machines and one controller server +10 enslaved person binding machines. The FP-growth algorithm is implemented on a single-node machine [17]. The T1014D100K data in the Frequent Itemset database was selected as the test data. This data set has 100,000 records. The minimum support levels were 2%, 4%, 6%, 8% and 10%. The test results are shown in Figures 4 and 5.

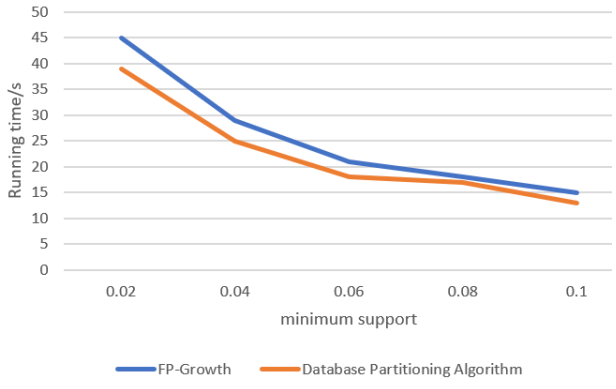

Figure 4. Experiment 1 has five node machines

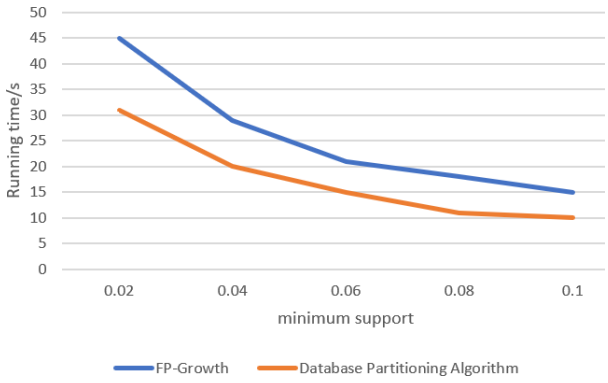

Figure 5. Experiment 1 has 10 node machines

- Compared with the conventional FP-extended algorithm, this method has a higher running speed. At the same time, with the increase of the number of node machines, the calculation time spent by this method also decreases [18]. At the same time, it also shows the superiority of parallel operation.
- The larger the minimum support value of the two methods, the shorter the search time. This is due to a higher minimum support value resulting in a lower frequency. For the FP-growth algorithm, the larger the minimum support value, the fewer FP trees are obtained. At the same time, recursive mining in the FP tree takes less time. For splitting databases, the larger the minimum support value, the fewer projective databases will be generated. The speed at which information is released becomes much faster. It also takes less time to explore the projected image database.
- Compared with the FP-extended algorithm, the acceleration rate of the segmentation library algorithm does not increase linearly with the increase of minimum support value. This is due to the significant time ratio between image database generation and distribution in the segmentation database algorithm. This kind of calculation does not exist in the FP-growth algorithm [19]. Therefore, when the minimum support gradually increases, the FP tree generated by the FP-growth algorithm becomes smaller and smaller, and the recursive mining time will become closer and closer to the linear decreasing trend.

6.2. Test 2

In a cluster system with one master computer and ten agent nodes, database segmentation is operated. Currently, the FP-growth method only applies to a single-node machine. The test results were generated using the IBMQUEST market Sample synchronization

data generator. Select a transaction database with 1,000,000 records. The minimum support is also set at 2%, 4%, 6%, 8% and 10%, respectively. The results of the experiment are shown in Figure 6. An algorithm that splits the transaction database as the size of the transaction database increases during execution is more efficient. The FP-growing FP tree becomes more considerable as the FP-growth algorithm is added. Recursive mining of many FP trees will take a lot of time. For the segmented database algorithm, it takes less time to generate and propagate. In addition, since the size of the divided sublibraries is similar, the load distribution on each node is more balanced. The sense of efficiency brought about by parallel computing through multiple-node machines is even more apparent.

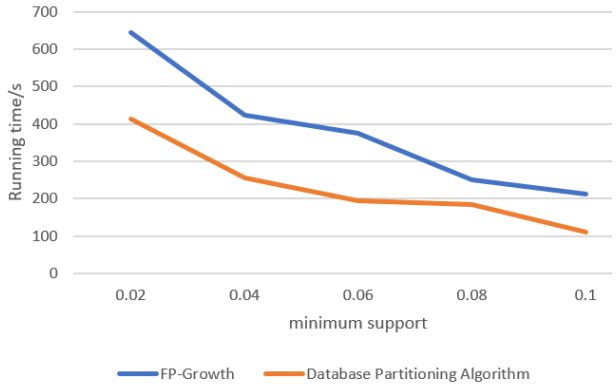

Figure 6. 10 node machines in experiment 2

7. Conclusion

The method given in this paper is based on the FP-growth method. This method adopts Hadoop architecture and Map Reduce design patterns. This paper proposes a new method based on a total frequency 1 projection library. The image library is then segmented. Because users can freely set the number of transaction records in each subdatabase, the FP subtree generated by each subdatabase can be effectively controlled. The problem that the algorithm is invalid because a single computer cannot store FP-tree is solved effectively. At the same time, the parallel generation and mining of FP can be solved well by using the programming pattern of Map Reduce. The size of the sublibrary that needs to be assigned to each node machine is equal. Thus, the load of each node machine is more balanced. When the proposed method is applied to large machines, massive data can be processed effectively.

Funding.

This paper was supported by Research on key Technologies of Intelligent Cloud Broadcasting System, Henan Science and Technology Department, a key R&D and promotion project in Henan Province in 2023 (tackling key scientific and technological problems). Project number: 232102210037.

References

[1] Putera, P. B., Suryanto, S., Ningrum, S., Widianingsih, I., & Rianto, Y. Using convergent parallel mixed methods and datasets for science, technology, and innovation policy dynamics research in Indonesia. ASEAN Journal on Science and Technology for Development,2022; 39(2): 61-68.

[2] Santoso, M. H. Application of Association Rule Method Using Apriori Algorithm to Find Sales Patterns Case Study of Indomaret Tanjung Anom. Brilliance: Research of Artificial Intelligence,2021; 1(2):54-66.

[3] Stylos, N., Zwiegelaar, J., & Buhalis, D. Big data empowered agility for dynamic, volatile, and time-sensitive service industries: the case of tourism sector. International Journal of Contemporary Hospitality Management,2021; 33(3):1015-1036.

[4] Mabrouki, J., Azrour, M., Dhiba, D., Farhaoui, Y., & El Hajjaji, S. IoT-based data logger for weather monitoring using arduino-based wireless sensor networks with remote graphical application and alerts. Big Data Mining and Analytics,2021; 4(1): 25-32.

[5] He, Q., Borgonovi, F., & Suárez‐Álvarez, J. Clustering sequential navigation patterns in multiple‐source reading tasks with dynamic time warping method. Journal of Computer Assisted Learning,2023; 39(3):719-736.

[6] Kh-Madhloom, J. Dynamic Cryptography Integrated Secured Decentralized Applications with Blockchain Programming. Wasit Journal of Computer and Mathematics Science,2022; 1(2): 21-33.

[7] Sunhare, P., Chowdhary, R. R., & Chattopadhyay, M. K. Internet of things and data mining: An application oriented survey. Journal of King Saud University-Computer and Information Sciences, 2022;34(6): 3569-3590.

[8] Haoxiang, W., & Smys, S. Big data analysis and perturbation using data mining algorithm. Journal of Soft Computing Paradigm (JSCP),2021; 3(1): 19-28.

[9] Nahar, K., Shova, B. I., Ria, T., Rashid, H. B., & Islam, A. S. Mining educational data to predict students performance: A comparative study of data mining techniques. Education and Information Technologies, 2021;26(5): 6051-6067.

[10] Arun, S., & Sudharson, K. DEFECT: discover and eradicate fool around node in emergency network using combinatorial techniques. Journal of Ambient Intelligence and Humanized Computing, 2023;14(5):5995-6006.

[11] Nasereddin, M., ALKhamaiseh, A., Qasaimeh, M., & Al-Qassas, R. A systematic review of detection and prevention techniques of SQL injection attacks. Information Security Journal: A Global Perspective,2023; 32(4): 252-265.

[12] Patro, K. K., Jaya Prakash, A., Jayamanmadha Rao, M., & Rajesh Kumar, P. An efficient optimized feature selection with machine learning approach for ECG biometric recognition. IETE Journal of Research,2022; 68(4): 2743-2754.

[13] Hasan, B. M. S., & Abdulazeez, A. M. A review of principal component analysis algorithm for dimensionality reduction. Journal of Soft Computing and Data Mining,2021; 2(1): 20-30.

[14] Garg, S., Sinha, S., Kar, A. K., & Mani, M. A review of machine learning applications in human resource

management. International Journal of Productivity and Performance Management,2022; 71(5): 1590-1610.
[15] Nandy, A., Duan, C., & Kulik, H. J. Using machine learning and data mining to leverage community knowledge for the engineering of stable metal–organic frameworks. Journal of the American Chemical Society,2021;143(42):17535-17547.
[16] Rehman, A., Naz, S., & Razzak, I. Leveraging big data analytics in healthcare enhancement: trends, challenges and opportunities. Multimedia Systems,2022; 28(4): 1339-1371.
[17] Xin, W. A. N. G., Zi-Yi, W. A. N. G., Zheng, J. H., & Shao, L. I. TCM network pharmacology: a new trend towards combining computational, experimental and clinical approaches. Chinese Journal of Natural Medicines,2021; 19(1): 1-11.
[18] Chen, X., Zou, D., Xie, H., & Cheng, G. Twenty years of personalized language learning. Educational Technology & Society, 2021;24(1): 205-222.
[19] Alzamily, J. Y. I., Ariffin, S. B., & Abu-Naser, S. S. Classification of Encrypted Images Using Deep Learning–Resnet50. Journal of Theoretical and Applied Information Technology, 2022;100(21): 6610-6620.