An Improved Sequential Pattern Mining Algorithm Based on Data Ming

Lili Wang

School of Information Engineering, Harbin University, Harbin 150001, China

108587570@qq.com

Keywords: Sequential pattern mining; Projection database; PrefixSpan; Large datasets; Map-Reduce

Abstract. This paper improves PrefixSpan algorithm and proposes ISPM (Improved Sequential Pattern Mining) Algorithm. This algorithm can greatly reduce the numbers of construction projection database, thus improving the efficiency of sequential pattern mining. In addition, the algorithm proposes the concept of sequential pattern values, and reorders the results of the mining sequence patterns by the values of sequence pattern, so that it can find the most important sequence patterns. Then we make experiments to verify the efficiency of ISPM algorithm, from different supports, different types of datasets and different sizes of datasets. Propose the ISPM of Map-Reduce algorithm. In practical applications, in the face of huge datasets, the efficiency of the ISPM algorithm is facing bottlenecks. Therefore, we propose ISPM of Map-Reduce algorithm. By way of distributed processing, we put large tasks into multiple smaller tasks, then do sequence pattern mining in parallel on each name-node. Then we make experiments to verify the efficiency of the algorithm. The first experiment is to verify the speedup of the algorithm between single platform and Hadoop. The second experiment is to test the efficiency of the algorithm in different sizes of the datasets. From two experiments, we can find that this algorithm could be able to improve the efficiency in the face of large datasets.

Introduction

Data mining technology can not only process tremendous historical and existing data, and also discover valuable information from such huge historical data, providing guidance to the practical production, operation and development [1]. Sequence pattern mining, as a significant research topic in the data mining field, is a knowledge discovery process in which frequent sub-sequence is found from sequential database to use as pattern. Rakesh Agrawal and Ramakrish [2] et al. firstly proposed sequence pattern mining algorithm Apriori based on data analysis of shopping basket. Sequence pattern mining is of practicability and easy comprehension so that it gains wide concern and deep investigation. Through researches in recent years, some typical sequence pattern mining algorithms have been generated [3,4,5]. They improved to a certain degree the efficiency of data exploration; however with the age of big data approaching, the scale of data set has been becoming bigger and bigger in more and more complicated structure [6,7,8]. When traditional sequence pattern mining algorithm is doing data mining, the existence of enormous irrelevant and redundant data increases space-time costs and sometimes causes memory overflow, weakening greatly the performance of traditional sequence pattern mining technique. Besides, sequence patterns excavated by the traditional algorithm are of low quality, unable to meet customer's actual requirements, no useful information mined. When it enters into the era of big data, the commonest issue facing lots of enterprises is "vast information but poor information". How to dig out the most valuable information from massive big data collection has become a research focus at present and an issue to be solved urgently. At present, the research of sequential pattern mining can be divided into the following categories: basic sequential pattern mining, incremental sequential pattern mining, multidimensional sequential pattern mining, constraint based sequential pattern mining.

Incremental sequential pattern mining: It is mainly used to study how to maintain sequential patterns and improve the efficiency of data mining. Typical algorithms are: ISM algorithm, ISE

  DOI: 10.25236/icess.2019.236

algorithm, IUS algorithm. The ISM algorithm uses the vertical data storage mode to construct the pattern of the incremental sequence lattice. When you are in a dataset, you scan only once, and then add the result to the incremental sequence. The ISE algorithm adds the incremental database sequence to the original database sequence and takes it as the suffix of the original sequence until all the candidate sequence patterns are generated. IUS algorithm uses extended prefix and suffix, using the reverse boundary in ISM, defines the minimum number of support for the reverse sequence. Multidimensional sequential pattern mining: it is the fusion of multi-dimensional valuable information into a single dimensional sequence, and then mining. The most valuable information, so as to meet the needs of reality [9,10].

Considering weakness of the traditional sequence pattern mining algorithm PrefixSpan, we improved it and proposed ISPM algorithm. Through experiments, it proves that ISPM algorithm performs better than PrefixSpan. On the basis of studying sequence pattern mining, and through improvement of PrefixSpan algorithm, we proposed ISPM algorithm based on Map-Reduce and used it in real big dataset. Valuable information is discovered from big dataset and used to guide practical business activities.

Related research theories of methodology

PrefixSpan algorithm. Take the thought of "divide and conquer"; firstly scan sequence database, find all sequence patterns whose length is 1 and put those sequence patterns as prefix; divide sequence database into a few small projection database; then do recursive sequence pattern mining in each projection database; first of all it needs a sequence database S, which produces several projection database S1， S2….Sn according to prefix division; then do recursive mining in those project database till all frequent sequence patterns are found.

Problems with PrefixSpan algorithm. Although PrefixSpan algorithm can increase mining efficiency, it still has shortcomings: 1. It needs construct big project database and constructing such database costs huge. 2.It requires scanning projection database in a recursive manner, consuming big space-time cost and reducing the algorithm's mining efficiency.3. Mined frequent sequence patterns are ranked in lexicographical order, unable to meet real needs.

Improved Sequential pattern mining algorithm. ISPA algorithm was improved from the two aspects: 1. It presents the way combining interlayer projection and pruning strategy, considerably reducing the number of project database and the time for scanning project database, thus the efficiency is enhanced a lot; interlayer projection strategy refers to: construct project database in an original manner when sequence pattern whose length is odd number is mined; and construct a lower triangular M matrix rather than project database when sequence pattern whose length is even number is discovered, declining largely the number of constructed project database; pruning method removes directly from sequence database the sequence pattern whose support is less than the minimum by setting certain constraint condition. 2. For mined frequent sequence patterns, calculate the value of each sequence pattern in terms of support degree and weighted value; then rank order according to sequence pattern value; in most sequence pattern mining, every item and each frequent sequence pattern in the sequence database is regarded equally important; the result of sequence pattern mining is sorted in lexicographical order; however in real life, they're not equal; each item in the sequence represents different business and their importance is of different level; for users, they concern only significant business; hence restrictive condition is made, which considers significance of every item and regards it as weight to add in sequence pattern mining, as to get mining result which is more interesting to users.

Relative definitions and theorem. S matrix: assume α is a sequence pattern whose length is L. $\alpha_1^{'}, a_2^{'}, a_m^{'}$ is sequence patterns at length of Length-L-1 with the prefix α in project database; then S matrix of α - project database is put as $M(\alpha_i^{'}, a_j^{'}), (1 < i < j < m)$.

It's defined as follows:

1. $M(\alpha_i^{'}, a_j^{'})$ contains one counter; if the last element of $\alpha_i^{'}$ is only one item x, i.e. $\alpha_i^{'} =< ax >$, the counter records the occurrence number of sequence $< \alpha_i^{'}, x >$ in α 's project database; otherwise, the counter records 0;

2. $M(\alpha_i^{'}, a_j^{'}), (1 < i < j < m)$ is form of (A, B, C), where A, B, C are three counts;

Item's weighted value: this value is used to represent non-negative real number of every item's importance in the sequence database; in order to show the importance of every item, we treat item attribute in the sequence database as item's weighted value . Data standardized transformation: zoom data by scale to make them in a small range; not limited by unit, data are transformed to non-dimensional pure value, good for us to compare or do weighting treatment of indicators of different magnitudes or units; Min-max standardization is linear conversion of original data with result mapped into [11]. The conversion function is shown in Equation (1).

$$x^* = \frac{x - \min}{} \tag{1}$$

Where, x^* represents the converted value. x^* represents the original data value. Min represents the minimum value of the sample data. Max represents the maximum value of the sample data. Weight value of frequent item sets: In the sequence database S, if P is a frequent sequence, and $P =< S_1 S_2 ... S_i >$, then the frequent item set P's weight value is equal to the frequent item sets the weight accumulation and / frequent item set length. It is shown in Equation (2).

$$Weight(P) = \sum^{length(p)} Weight(P) / lengh(P) \tag{2}$$

Where, Length (P) is the length of the frequent sequence P. Weight (P) is the weight of frequent sequence P. Sequence pattern value VSP (P): the sequence mode value is used to measure the importance of frequent sequences, the main reference weight value and the support of the two indicators. The sequence mode value VSP is equal to the weight value * support. It is shown in Equation (3).

$$VSP(P) = Weight(P) * Support(P) \tag{3}$$

ISPM algorithm based on Map-Reduce

Basic ideas.1. Data fragmentation: spread big dataset and store in many nodes to alleviate storage pressure of each node, and meanwhile overcome bottleneck of memory; 2. Parallel counting: scan globally database once; with Map-Reduce model, do parallel counting of every sequence's support in each node; then combine results of all nodes to get frequent itemset Flist whose global length is 1; in Flist, if the sequence support is smaller than minimum support minSup, prune it; save in the Flist1 all sequences whose sequence support is not below minSup; 3.Construct triangular matrix: use n sequences in the Flist1 as axis X and Y to construct one n*n lower triangular M matrix; if its sequence support is below minimum support minSup, prune it; save in the Flist2 all sequences with support not smaller than minimum support; 4. Equal grouping: utilize load balancing strategy to divide all sequences in Flist2 into Q groups; each group has one group number named $gid_1 (1 <= i <= q)$; each group contains numerous items, which are organized to a new list Glist; 5.Parallel mining: according to Glist, divide big dataset into Q groups; group number of dataset conforms to Glist group number; then in each group, dig out concurrently all items' frequent sequence patterns; the parallel mining process is completed by the second pair of Map-Reduce.

Data fragmentation. Big dataset is automatically split into a few consecutive parts, which are then respectively stored on different nodes. Each divided part is called data fragmentation. The dividing process is completed by Hadoop, copying dataset to HDFS. Hadoop framework can automatically finish data fragmentation, that's because Hadoop's HDFS system can automatically separate input files into several Blocks and save them on different nodes.

Parallel counting. Do global scanning of dataset. According to Map-Reduce model, compute support of every sequence on each node; then merge results of all nodes to get frequent itemset Flist1 with global length 1. Here we use a pair of Map-Reduce to complete the job, where each Map function corresponds to every data fragment; Map function's input is transaction of every data fragmentation; output is sequence and sequence's support counting. Reduce function is responsible to combine and sort in order sequences of all Map outputs. For the same sequence, calculate the sum of sequence support.

The first pair of Map-Reduce includes Map end and Reduce end. Map end input format is (key=Num,value=T), and output format is (key=item,value=1). Reduce end input format is (key=item,values = {1,1,1,1,1…}) and output format is Reduce (key=item value=itemCount), where Num refers to number of data fragmentation; T is transaction of relative data fragmentation; Item stands for every item appearing in T; value is numerical frequency of an item appearing; ItemCount is total number of relative item appearing, i.e. support count.

Algorithm1 Map end of the pseudo code

Input: key is the number of each shard， value is T.

Output: <key=item，value=1>， where key is the item of T ，value is the number of item

```
1. Begin;
2. String str= value.toString();
3. While(str. hasNext());
4. { Item=str.next();
5. Context. Write(item,1)};
6. End
```

Algorithm2 Reduce end of the pseudo code

Input: key is the item of T

Output: < key=item，value=itemCount >, where key is the item of T ,value is itemCount

```
1. Begin;
2. For (val: values);
3. {sum=sum + val};
4. If (sum >= min_sup)
5. {Context. Write (key, sum)};
6. End
```

Equal grouping

The purpose for grouping is to partition database business according to Flist2, dividing big dataset evenly into several small dataset and distributed to different nodes; then do pattern mining in a concurrent manner. Whether equal division is enabled affects directly every dataset load which is balanced or not in the next step and the parallel efficiency of the overall algorithm. Therefore before grouping, it needs to estimate the load of each group. We use the recursive times of all items'

condition pattern tree contained in each group as the load of this group. Since during division, it's not possible to figure out the recursion times of each item; so it's required to estimate the mining load of every item. To solve that problem, we suggest using the maximum value of the longest path of every item's corresponding condition pattern tree as the position of the item in Flist2 .If the greatest value of the longest path is n, the biggest recursive times for mining the item is $n-1+n-2+...+1=(n*(n-1))/2$. Thus we get the estimation of every item's mining load as $n*(n-1)/2$.

The division is made in following steps:

1. Calculate load of each item in Flist2 and rank load of every item into Qlist in a descending order, divided artificially into Q groups;
2. Use q items in Qlist as the first q items in Glist; each group corresponds to one item; each group's load includes total loads of all items;
3. Upload the biggest item which is not grouped in Qlist to the group with minimum load value and incorporate the load value of the item to form new load value of the group;
4. Repeat (3) till all items in Qlist completes grouping;
5. Finally save Glist in HDFS as for other nodes to be able to share the group.

Parallel mining

Divide big dataset into Q groups according to Glist. The group number of big dataset keeps in line with Glist group number, which can ensure that all groups are separate and mutually independent. That is to say items contained in all groups are not the same, besides the resultant frequent sequence of each group differs from each other. Next in every group, discover concurrently all items' frequent pattern mode. This mining process is completed by the second pair of Map-Reduce. In it, Map function is responsible to group dataset based on the division of Flist2 to guarantee each acquired dataset mutually separate and independent from each other; Reduce function is responsible for doing sequence pattern mining of well-grouped dataset. Parallel mining includes Map end and Reduce end.

Experiment Design and Discussion

Experimental environment. Based on cloud computing platform Hadoop cluster, Hadoop version number is 0.20.2. A total has 4 nodes. A master control node (Master), 3 data nodes (Slaves), the main node configuration: CPU is the six core, E5-2430 Xeon, memory 64G, hard disk 2T*8, operating system for Centos 6. Each data node configuration: 4G memory, 500GB, Gigabit Network card, the operating system is Ubuntu10.04. Testing dataset used for the experiment came from open source data mining platform SPMF. We chose three bigger testing dataset: chess, Pumsb, T10I4D100K, which are passed to HDFS for respective testing in two experiments. One is to validate the algorithm's speedup ratio in Hadoop platform, comparing the mining performance in single-machine case and hadoop environment; the other is to verify mining efficiency of ISPM algorithm with different data sets based on Map-Reduce.

Analysis of speed-up ratio of the algorithm on single machine and Hadoop platform. Speed-up ratio is defined as the percentage of consumed time by single processor system and parallel processor system when they're running the same task. It measures the performance of parallel system or program parallelization. Speed-up ratio Equation (4):

$$Sp=T1/Tp \tag{4}$$

In it Sp refers to speed-up ratio; $T1$ indicates time consumed by single processor; Tp is time consumed by parallel system with P processors; when $Sp=p$, it's called linear speed-up ratio. Under we use Pumsb dataset as testing dataset and implement ISPM algorithm on single machine and Hadoop cluster based on Map-Reduce to test its speed-up ratio. It is shown in Figure1.

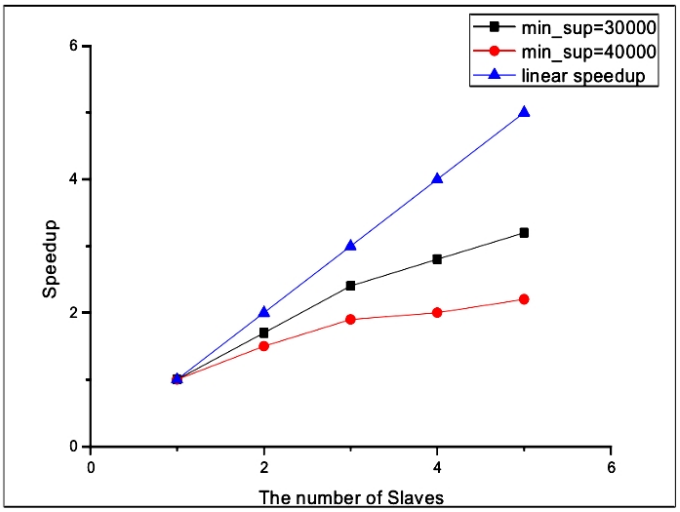

Figure 1 The acceleration ratio based on the ISPM- Map-Reduce algorithm in the data set pumsb

Figure 1 displays when dataset being processed is the same, speed-up ratio varies with different support; besides when the number of cluster Slaves is increasing, speed-up ratio tends to rise as well; comparatively to single node, multi-node data mining is more efficient, because it employs BLSPM algorithm based on Map-Reduce. Through distributive processing, every one task is decomposed to several smaller tasks then do parallel processing of each smaller task, which helps improve mining efficiency.

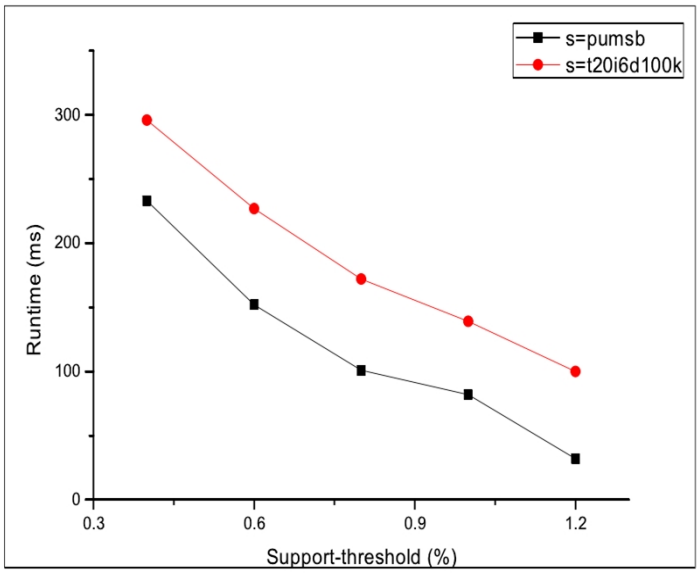

Figure 2 Running time comparison of based on Map-Reduce ISPM algorithm in two data sets

Algorithm analysis with dataset of different size. To validate better Map-Reduce algorithm's mining efficiency in dataset of different size, we chose dataset of two different orders of magnitudes to do experiments. Two data sets are Pumsb which include 49046 records and T20I6D100k which includes 100000 records. They both came from open source data mining platform SPMF, commonly used for sequence pattern mining. We carry out ISPM algorithm based on Map-Reduce with the two data sets. It is shown in Figure2.

Figure 2 shows that ISPM algorithm based on Map-Reduce works more efficient in big dataset than small one, owing that ISPM algorithm based on Map-Reduce is processed in blocks, with default data block size 64M. If dataset is too small, not big enough to be one data block, it's processed like

one data block. That will easily cause small data blocks to consume computing capability of default data block; while big dataset is far bigger than one data block, which can make full use of Map block to deal with tasks. Thus we think ISPM algorithm based on Map-Reduce works better in mining big dataset and has advantage.

Conclusions

In light of the problem of less efficiency in mining big dataset, the paper presented ISPM algorithm based on Map-Reduce. Through two groups of experiments, it confirmed the performance of ISPM algorithm based on Map-Reduce in these two points: The proposed method achieved higher speed-up ratio on multi-node Hadoop than single machine, it worked more efficiently in mining big dataset than small one.

References

[1] L.M. Aouad, T.M. Kechadi, "Distributed Frequent Item sets Mining in Heterogeneous Platforms", Engineering, Computing and Architecture, 2007

[2] R. Agrawal, R. Srikant, "Mining Sequential Pattern", Proc of the 11 International Conference on Data Engineering. Taipei, 1995.

[3] X.G. Dong, "An Effective Algorithm for Mining Weighted Sequential Pattern Mining based on Graph Traversal", Control and Decision, 5, 663-669,2009

[4] W. Gong, P. Liu, "Sequential Pattern Mining based on Improved PrefixSpan Algorithm", Computer Application, 9,2405-2407, 2011

[5] F. Hu, "Analysis and Improvement of Frequent Sequential Pattern Mining Algorithm", Journal of Qinghai Normal University (NATURAL SCIENCE EDITION), 3, 03:35-38,2009

[6] L. L. Wang, J. Fan, "An Improved Algorithm for Mining Sequential Pattern Mining Based on PrefixSpan", Computer Engineering, .23,56-58,2013

[7] F. Lv, W. W. Zhang, "Research on the Characteristics of 4 Kinds of Sequential Pattern Mining Algorithms", Journal of Wuhan University of Technology, 2, 57-60,2006

[8] T. S. Li, W. Wang, "Research on Mining Algorithm of Web Access Sequence Patterns", Computer Science, 12, 41-44,2013

[9] W.Z. Lin, "the New Web Sequential Pattern Mining Algorithm", Journal of Xiamen University (NATURAL SCIENCE EDITION), 1, 25-31,2013

[10] H. Wang, D Ding, "Research and Development of Sequential Pattern mining", Computer Science, 36 (12): 14-17, 2009

[11] Q. Q. Jiang, X. Wang, S.C. Huang, "An Improved Frequent Pattern Mining Algorithm in MNWAP-mine", Journal of Jiangsu University of Science and Technology (NATURAL SCIENCE EDITION), 1,59-64,2016