The Algorithm for Mining Global Frequent Itemsets based on Big Data

He Bo
School of Computer Science and Engineering
Chongqing University of Technology
Chongqing, China
E-mail: heboswnu@sina.com

Abstract—There were some algorithms for mining global frequent itemsets. Most of them adopted apriori-like algorithm, so that a lot of candidate itemsets were generated. To solve the problems, the algorithm for mining global frequent itemsets based on big data was proposed, namely, MGFI algorithm. MGFI algorithm computed local frequent itemsets by mapreduce, then the center node collected data, finally, global frequent itemsets were got by mapreduce. MGFI algorithm required less communication traffic by the searching strategies of top-down and bottom-up. Theoretical analysis and experimental results suggest that MGFI algorithm is fast and effective.

Keywords- Data Mining; Global Frequent Itemsets; Big Data; Mapreduce; FP-tree

I. INTRODUCTION

There are some distributed data mining algorithms [1], such as CD [2] and FDM [3]. Most of them adopt Apriori-like algorithm, so that a lot of candidate itemsets are generated and the database is scanned frequently. This causes heavy communication traffic among the nodes. To solve these problems, this paper proposes the algorithm for mining global frequent itemsets based on big data, namely, MGFI algorithm.

Big data is a broad term because the data sets are so large or complex that the traditional data processing applications are inadequate. Challenges include analysis, capture, data curation, search, sharing, storage, transfer, visualization, and information privacy. The term often refers simply to the use of predictive analytics or other certain advanced methods to extract value from data, and seldom to a particular size of data set. Accuracy in big data may lead to more confident decision making. And better decisions can mean greater operational efficiency, cost reductions and reduced risk.

Big data is being generated by everything at all times. Every digital process and social media exchange produces it. Systems, sensors and mobile devices transmit it. Big data is arriving from multiple sources at an alarming velocity, volume and variety. To extract meaningful value from big data, you need optimal processing power, analytics capabilities and skills.

MapReduce is a programming model and an associated implementation for processing and generating large data sets with a parallel, distributed algorithm on a cluster.

Mapreduce has large scale and highly scalable. The massive distributed date mining based on big data is a very important field.

II. RELATED DEFINITION AND THEOREM

A. Description of Mining Global Frequent Itemsets

The global transaction database is DB, the total number of tuples is M. Suppose P1, P2,…, Pn are n nodes, node for short, there are Mi tuples in DBi, if DBi (i=1,2,…,n) is a part of DB and stores in Pi, then $DB = \bigcup_{i=1}^{n} DB_i$, $M = \sum_{i=1}^{n} M_i$.

Mining global frequent itemsets can be described as follows: each node Pi deals with local database DBi, and communicates with other nodes, finally, global frequent itemsets of global transaction database are got.

B. Related Definition

Definition 1 For itemsets X, the number of tuples which contain X in local database DBi(i=1,2,…,n) is defined as local frequency of X, symbolized as X.si .

Definition 2 For itemsets X, the number of tuples which contain X in global database is global frequency of X, symbolized as X.s .

Definition 3 For itemsets X, if X.si $\geq$ min_sup*Mi(i=1,2,…,n), then X are defined as local frequent itemsets of DBi, symbolized as Fi. min_sup is the minimum support threshold.

Definition 4 For itemsets X, if X.s $\geq$ min_sup*M, then X are defined as global frequent itemsets, symbolized as F.

C. Related Theorem

Theorem 1 If itemsets X are local frequent itemsets of DBi, then any nonempty subset of X are also local frequent itemsets of DBi.

Corollary 1 If itemsets X are not local frequent itemsets of DBi, then the superset of X must not be local frequent itemsets of DBi.

Theorem 2 If itemsets X are global frequent itemsets, then X and all nonempty subset of X are at least local frequent itemsets of a certain local database.

Theorem 3 If itemsets X are global frequent itemsets, then any nonempty subset of X is also global frequent itemsets.

Corollary 2 If itemsets X are not global frequent itemsets, then superset of X must not be global frequent itemsets.

D. *FP-tree and FP-growth Algorithm*[4]

Definition 5 FP-tree is a tree structure defined as follow.

1) It consists of one root labeled as "null", a set of itemset prefix subtrees as the children of the root, and a frequent itemset header table.

2) Each node in the itemsets prefix subtree consists of four fields: item-name, count, parent and node-link.

3) Each entry in the frequent-item header table consists of three fields: i,Itemname. ii, Side-link, which points to the first node in the FP-tree carrying the item-set. iii, Count, which registers the frequency of the item-name in the transaction database.

FP-growth algorithm adopts a divide-and-conquer strategy. It only scans the database twice and does not generate candidate itemsets. The algorithm substantially reduces the search costs. The study on the performance of the FP-growth shows that it is efficient and scalable for mining both long and short frequent patterns, and is about an order of magnitude faster than the Apriori algorithm.

III. MGFI ALGORITHM

A. *Design Thoughts of MGFI Algorithm*

MGFI sets one node P0 as the center node, other nodes compute local frequent itemsets with FP-growth algorithm and mapreduce. Then the nodes send local frequent itemsets Fi to the center node P0. P0 gets local frequent

$$F' = \bigcup_{i=1}^{n} F_i$$

itemsets F'( ) which are pruned by the searching strategies of top-down and bottom-up. P0 sends the remain of F' to other nodes. For local frequent itemsets $d \in$ the remain of F', P0 collects local frequency d.si of d from each node and gets global frequency d.s of d. Global frequent itemsets are gained by mapreduce.

F' are pruned by the searching strategies of top-down and bottom-up which adopted one after another. Pruning lessens communication traffic.

The searching strategy of top-down is described as follow.

Confirming the largest size k of itemsets in F'.

2) Collecting global frequency of all local frequent k-itemsets in F' from other nodes Pi .

3) Judging all local frequent k-itemsets in F', if local frequent k-itemsets Q are not global frequent itemsets, then Q are deleted from F', else turn to 4).

4) Adding Q and any nonempty subset of Q to global frequent itemsets F according to theorem 3 . Deleting Q and any nonempty subset of Q from F' .

The searching strategy of bottom-up is described as follow.

1) Collecting the global frequency of all local frequent 2-itemsets in F' from other nodes Pi.

2) Judging all local frequent 2-itemsets in F', if local frequent 2-itemsets R are global frequent itemsets, then R are Added to global frequent itemsets F and R are deleted from F' , else turn to 3).

3) Deleting R and any superset of R from F' according to Corollary 2.

Each node adopts FP-growth algorithm to compute local frequent itemsets in MGFI. Adopting FP-tree structure and mapreduce, FP-growth algorithm greatly reduces database scanning times and runtime compared with Apriori-like algorithm.

B. *Description of MGFI Algorithm*

The pseudocode of MGFI is described as follows.

Alogrithm MGFI

Input: The local transaction database DBi which has

$$M = \sum_{i=1}^{n} M_i$$

Mi tuples and , n nodes Pi(i=1,2,…n), the center

node P0, the minimum support threshold min_sup.

Output: The global frequent itemsets F.

Methods: According to the following steps.

step1. /*each node adopts FP-growth algorithm and mapreduce to produce local frequent itemsets*/

for(i=1;i<=n;i++) /*gaining global frequent items by mapreduce*/

{Scanning DBi once;
computing local frequency of local items Ei ;
Pi sends Ei and local frequency of Ei to P0;
}

P0 collects global frequent items E from Ei;

E is sorted in the order of descending support count;

P0 sends E to other nodes Pi; /*transmit global frequent items to other nodes Pi */

for(i=1;i<=n;i++)

{creating the FP-treei; /*FP-treei represent FP-tree of DBi */

F_i =FP-growth(FP-treei, null);

}

Step2./* P0 gets the union of all local frequent itemsets and prunes*/

for(i=1;i<=n;i++)

Pi sends F_i to P0; /* F_i represent local frequent itemsets of Pi */

$$F' = \bigcup_{i=1}^{n} F_i$$

P0 combines F_i and produces F' ; /* */

Pruning F' according to the searching strategy of top-down;

Pruning F' according to the searching strategy of bottom-up;

/*The searching strategies of top-down and bottom-up are described in section 3.1 */

P0 broadcasts the remain of F' ;

Step3./*computing global frequency of itemsets by mapreduce*/

for(i=1;i<=n;i++)

{ for each items $d \in$ the remain of F'

Pi sends d.si to P0; /*computing d.si To solve FP-treei */

}

for each items d∈the remain of F'

$\sum_{i=1}^{n} d.si$

d.s= ; /* d.s represents global frequency of itemsets d */

step4./*getting global frequent itemsets by mapreduce*/

for each items d∈the remain of F'

if (d.s>=min_sup*M)

F=F$\cup$ d;

IV. Comparison Experiments of MGFI

This paper compares MGFI with classical distributed algorithm CD and FDM. All tests are performed on 100M LAN, 6 PC as distributed nodes and 1 server as center node. The experimental data comes from the sales data in July 2012 of a supermarket.

Comparison experiment: It is a way of changing the minimum support threshold while adopting fixed number of nodes. MGFI compares with CD and FDM in terms of communication traffic and runtime. The results are reported in Fig .1 and Fig .2.

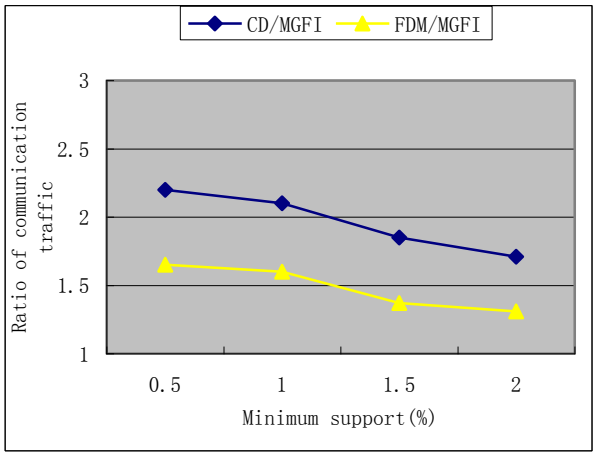

Figure 1. Comparison of communication traffic

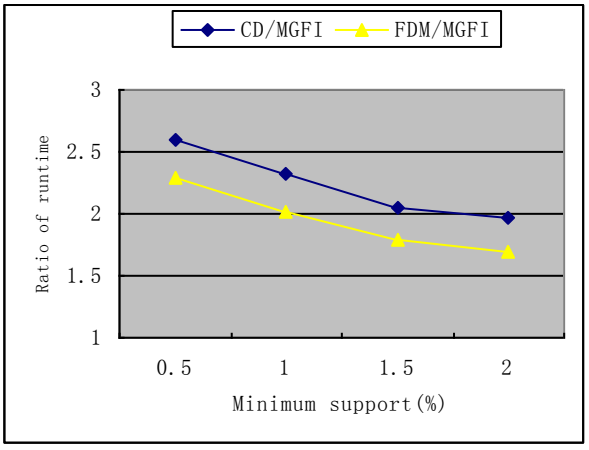

Figure 2. Comparison of Runtime

The comparison experiment results indicate that under the same minimum support threshold, communication traffic and runtime of MGFI decrease while comparing with CD and FDM.

V. Example of MGFI Algorithm

The database DB, as show in TABLE I. Min_sup is the minimum support threshold, min_sup=0.4.

TABLE I. DATABASE *DB*

database	*ID*	*Transaction*
DB	100	*a, b, c, k, m, f, e, l, p*
	101	*c, k, b, m, o, q*
	102	*a, b, c, d*

According min_sup=0.4, all frequent items can be got. All frequent items are sorted in the order of descending support count. As shown in TABLE II.

TABLE II. THE FREQUENT ITEMS AND SUPPORT COUNT

Frequent Items	Support count
c	3
a	2
k	2
b	3
m	2

All frequent items E={c, b, a, m, k},the FP-tree is constructed according to E, as shown in fig .1. According to Theorem 2, If item x is not frequent item, and {x}$\subseteq$X , then itemsets X must not be frequent itemsets. Hence the FP-tree only contains frequent items.

The frequent itemsets are computed by FP-growth algorithm and FP-tree. F={{c, b, a}, { c, b, m, k}, {c, b}, {c, a},{b, a}, { c, b, m }, { c, b, k },{ c, m, k }, { b, m, k }, {c, m} , {b, m}, {c, k}, {b, k}, {m, k}}.

VI. Conclusions

MGFI pruned by the searching strategies of top-down and bottom-up. The global frequent itemsets are gained by mapreduce. Theoretical analysis and experimental results suggest that MGFI is fast and effective.

Acknowledgment

This research is supported by the social science planning and cultivation project of Chongqing under grant No.2014PY50 and the humanities and social science research project of Chongqing municipal education commission under grant No. 15SKG131. This research is supported by the fundamental and advanced research projects of Chongqing under grant No. CSTC2013JCYJA40039 and the scientific and technological research program of Chongqing municipal education commission under grant No. KJ130825.

References

[1] Chen ZB, Han H, Wang JX. Data Warehouse and Data Mining[M].Beijing: Tsinghua University Press, 2009.

[2] Agrawal R, Shafer JC. Parallel mining of association rules[C]. IEEE Transaction on Knowledge and Data Engineering, 1996, 962-969.

[3] Cheung DW, Han JW, Ng WT, Tu YJ. A fast distributed algorithm for mining association rules[C]. In: Proceedings of IEEE 4th International Conference on Management of Data, Miami Beach, Florida,1996, 31-34.

[4] Han JW, Pei J, Yin Y. Mining frequent patterns without Candidate Generation[C]. In: Proceedings of the 2000 ACM SIGMOD international conference on Management of data, Dallas, Texas, United States,2000,1-12.

[5]He B, Yue W, Yang W and Yuan C. Fast Algorithm for Mining Global Frequent Itemsets Based on Distributed Database [C]. Rough Sets and Knowledge Technology, Chongqing, 2006, 415-420.

[6]He B. Fast Mining of Global Maximum Frequent Itemsets in Distributed Database [J]. Control and Decision, 2011,26(8):1214~1218. (in Chinese with English abstract)

[7]He B, He Y. Incremental Updating Algorithm of Global Maximum Frequent Itemsets in Distributed Database[J]. Journal of Sichuan University(Engineering Science Edition), 2012,44(3):112~117. (in Chinese with English abstract)

[8]Han JW, Kamber M, Pei J. Data Mining: Concepts and Techniques Third Edition [M]. San Francisco: Morgan Kaufmann, 2011.

[9] Buyya R, Yeo CS, Venugopal S. Market- Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities[C]. Proceedings of the 2008 10th IEEE International Conference on High Performance Computing and Communications, 2008: 5-13.

[10] Lin KW, Deng DJ. A novel parallel algorithm for frequent pattern mining with privacy preserved in cloud computing environments [J]. International journal of ad hoc and ubiquitous computing, 2010, 6(4): 205-215.

[11] Frank E. Gillett. Future View : The New Tech Ecosystems of Cloud, Cloud Services and Cloud Computing[J]. Forrester Report, 2008,6(1):59-62.

[12] Li W, Hen J, Pei J. CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules[C]. Proceedings of the 2001 IEEE Conference on Data Mining. San Jose, 2001:369-376.

[13] Liu B, Hsu W, Ma Y. Integrating Classification and Association Rule Mining[C]. Proceedings of the Fourth ACM SIGKDD Conference on Knowledge Discovery and Data

[14] He B. Fast Mining Algorithm of Association Rules Base on Cloud Computing[C]. EMEIT2012: 2209-2212.

[15] He B. The Algorithm of Mining Frequent Itemsets Based on MapReduce[C]. SCTEA 2013: 529-534.

[16] Tzeras K, Hartmann S. Automatic Indexing Based on Bayesian Inference Networks[C] . Proceedings of 16th ACM SIGIR Conference, 1993:22-34.