FREQUENT PATTERN MINING ON MESSAGE PASSING MULTIPROCESSOR BY UPDATINGTHE TABLE ENTRIES

M. Deeba
Loyola College, Chennai/Assistance Professor
Dr. Mary Immaculate Sheela
Pentecost University College, Ghana/ Professor

Abstract: Many algorithms have been developed to speed up mining performance on single core systems. But, when the text size is very large, it requires more memory space and computational cost are extremely high. So the trend goes with text mining with multi core machines. To find frequent patterns in text, the algorithms like Frequent Keyword mining(FKM) and FP growth are used. In our proposal, we include the multi core machines for text mining and introduce a new concept for finding the frequent patterns using message passing mechanism among the cores by updating the table entries. The new introduced technique is very simple and it requires less memory space than FKM and FP growth algorithm

Keywords: Parallel FP-Growth, Frequent Keywords Mining, Multi core Systems.

Introduction: Text mining is an important area for information extraction from huge dataset. The performance of Text mining on single core is decreased when the text is large. So, in recent days, people are implementing the Text mining algorithms on multi core machines. In Text mining, the frequent data item are generated from large datasets by applying mining algorithms like Apriori and FP-Growth etc.. The FPM algorithms are applied in the field of Market analysis, sales, product placement, for promotions and in text searching. Also it is applied in the Medical field, smart home with sensors etc.. The Apriori algorithm generates keywords and tests if they are frequent. The subset checking and support counting are expensive with computation and I/O in Apriori. The FP-Growth algorithm is not generates keywords. FP-Growth built a data structure FP-Tree and extract frequent items from it. The FP-Tree uses too much of pointers which are used in between the nodes containing the same item. The major disadvantage of FP- Tree is, which need huge amount storage as the original data or more. The worse case becomes when every transaction has a unique set of item. Also, the construction of FP-Tree is very expensive. Even though FP-Growth is much faster than Apriori algorithm, because of above drawbacks FP-Growth algorithm is not flexible to use.

In this paper, a unified model is proposed to count frequent patterns using message passing among the cores by updating the table entries in them. We are using the F-List, which contains the keywords and related frequency count of each data item. In our proposed algorithm we are taking same F-Lists which is proposed by Krishna Gadia and Kiran Bhowmick, but setting code to each keywords with the frequency counts. F-List is common to all the cores. Introducing a new Table called Pattern Count-Table(PC-Table) which entries are unique in the cores. Using message passing technique the table entries are updated based on the frequency of the data items.

The paper is organized as follows: section1 is Introduction, section2 gives the related works, section3 describes the background that we are going to use, section4 contains the proposed solution, section5 is application in Trend Analysis and section5 is the conclusion of our study.

Related Work: Lot of research studies has been done in the field of Frequency Pattern Mining and several authors have proposed different algorithms. Even though, still there are many issues and challenges exist in this area. The Apriori algorithm, FP-Growth algorithm, Rapid Association Rule

Mining(RARM) and Equivalence CLass Transformation(ECLAT) etc.. are well known associated rule mining algorithms. Everyone of them are having its own advantages and disadvantages. The Apriori algorithm is widely used and easy to implement but it requires many database scans. The RARM is using a tree structure called SoTrielT is suitable for low transaction database. The APMS requires multiple passes over the database and data structure is required for maintaining large item sets is not specified. The FP-Growth algorithm is faster than other mining algorithms and it eliminate the repeated database scan. The major disadvantage of FP-Growth is the consuming of more memory.

Background:

Frequency-List(F-list) : Krishna Gadia and Kiran Bhowmick proposed the parallelization of FP-Growth algorithm in multi cores. They used FP-Growth algorithm and make slight modifications on it. On their first step of the algorithm, they used Natural Language Processing to extract keywords which are acting as the item sets can be associated with an id similar in the case of Transaction table. Then they construct Frequency-List(F-list) which contains the keywords that occur more often by setting threshold value. In the second step, they give transactions to cores keeping the load balance between the cores. The cores give the respective local FP-Trees. Then combining all the Partial FP-Trees from cores to obtain final FP-Tree. From Final FP-Tree the information is extracted based on the frequency of the patterns. In our proposal, We are going to use the F-List which is used by Krishna Gadia and Kiran Bhowmick, on our first step. By Adding another column in the F-List we are assigning code for each data.

Dictionary Based Coding: On our second step, instead of constructing FP-Tree on each core, we used a Table named as Pattern-Count-Table(PC-Table). For filling the table entries we are using the concepts based on the Dictionary based Coding/LempelZiv Welch(LZW) compression of lossless compression. LZW compression start with simple dictionary called string table by assigning code to each string, is similar to our F-List. Sample string table of LZW compression is given below.

Sample String Table:

Code	String
1	A
2	B
3	C

Figure 1

If the input string is "ABABBABCABABBA" the compression algorithm LZW works as in the figure 2.

S	C	Output	Code	String
			1	A
			2	B
			3	C
A	B	1	4	AB
B	A	2	5	BA
A	B			
AB	B	4	6	ABB
B	A			
BA	B	5	7	BAB

The LZW algorithm uses Compression and Decompression Algorithms. We applied the compression algorithm for the above string. In the Figure2, string patterns are compressed by assigning codes to them. With the given string "ABABBABCABABBA" start with first character A which is followed by second character B, this sort of pattern not in the list already. So it assigns a next code value 3 to them. Next, second and third characters are combined, this pattern is a new one so the next code value 5 is given to them. Next, third and Fourth characters A and B is combined which is already assigned by a

code. If it is, the entire string AB is taken for S. Next, the AB and B is combined assigned by a code and so on.

Proposed Solution: Our proposed algorithm uses Frequently-List and the basic works used in LZW Compression algorithm. Our F-List contains one more column called code. Each string in the F-List is assigned by a unique code. Usually the code starts from 1 as in the case of Sample string table of LZW. The F-List of above example with code values will be as follows.

F-List with Code Value:

String	Code	Frequency Count
A	1	7
B	2	5
C	3	5

Figure 3

For our algorithm, entries of figure 2 will be Changed as in the following figure4 which is named as Pattern-Count Table, Where F and S are representing the first and Second items. The next two columns are Code and frequency count respectively. The output and string columns in the figure2 table is eliminated. Instead of that adding new column frequency count.

Pattern-Count Table (PC-Table):

F	S	Code	Frequency Count
1	2	4	1+1+1
2	1	5	1+1
1	2		
2	2	6	1
2	1		
1	2		
2	3		7
Etc,.			

Figure 4

In the above figure4, all the PC-Table entries are filled with relevant codes of the patterns except the frequency count column. Where F and S represent First and Second strings. The frequency count of a string is increased by one at a time when the same string is repeated. Our problem of frequent pattern mining using updating the table entries are following the above method. Since we are using multi core systems, After the construction of F-List as in figure 3, we partitioned the datasets and give it to the cores keeping the load balanced among the cores. The cores are having local tables as in the figture4 and having the common F-List. The cores will perform their local computations and updating their table entries. Each time whenever a new pattern is found it is entered in the table and the new code will be assigned. But if it finds already existing pattern the new code will not be assigned but the core sends message to other cores with the code of the pattern. If the relevant pattern entry is in any one of the core it gives message to sending core that the pattern is already there. And the core which has the pattern will increased its relevant count by one. Note that here we are not constructing the FP-trees on each cores as in the case of Krishna Gadia and Kiran Bhowmick work. The final table will be constructed by merging the table entries from individual cores.

Application in Trend Analysis:
We are using the same application area example of Krishna Gadia and Kiran Bhowmick.

News feed	Keywords
OnePlus One Phones Sold in India Won't Receive OTA Updates: Cyanogen	OnePlus One, India, Cyanogen
OnePlus One is a Bit Too Big for a Phone	OnePlus One, Big Phone
No updates for Indian version of OnePlus One: Cyanogen	No updates, OnePlus One, Cyanogen
Cyanogen backpedals on promise to update OnePlus One phones inIndia	Cyanogen, backpedals, OnePlus One, India
OnePlus One bought via Amazon India won't get OTA updates, Cyanogen says	OnePlus One, India, Cyanogen
Cyanogen now says OnePlus One devices sold in India won't get updates	OnePlus One, India, Cyanogen
Cyanogen, Inc. looks to clarify OnePlus One update situation in India	Cyanogen, OnePlus One, India
Cyanogen says it will provide updates for OnePlus One in India	Cyanogen, OnePlus One, India
India Questions Roger Federer and Pete Sampras	India, Roger Federer, Pete Sampras
Roger Federer mesmerizes Delhi	Roger Federer, Delhi

Figure 5

The News feed are converted in to Keywords using Natural Language Processing. Taking first ten entries of the headlines and its keywords we are going to apply the model as discussed above. The F-List is prepared with code assignments as follows.

F-List with Code Value:

String	Code	Frequency Count
One plus One	**1**	**8**
india	**2**	**5**
Cyanogen	**3**	**3**
Roger Federer	**4**	**2**

Fig 6

Taking only 2 cores, we divide the ten keywords in to 5. The first 5 are given to core1, the rests are given to core2. Each core is having its own PC-Table. The PC-Tale for core1 is given bellow.

PC-Table- Core1:

F	S	Code	Frequency Code
1	2	5	1+1+1+1
5	3	6	1+1
1		1	1
1	3	7	1
3	1	8	1
8	2	9	1+1+1
1	2	5	
5	3	6	

Fig 7

PC-Table- Core2:

F	S	Code	Frequency Code
6			
6			
9			
9			
1	4	10	1
4			1

Fig 8

Starting from the entries of PC-Table of core1, the first row string entries are constructed from the keywords "OnePlus One, India, Cyanogen " which is given by NLP algorithm From F-List we are taking the code for "OnePlus One" is 1 and consecutive second string "India" is 2. Since this is a new pattern the new code 5 is given to the pattern and its corresponding frequency count is set as 1.Whenever a pattern is recognized on cores first it checks within its F and S values of each entry, if it exists, it simply increment the code value of the entry which is having the code already. If not it sends messages to other cores with F and S values checks with codes of other cores if the code is already in any of the cores, the core which is having that code will increment its corresponding frequency count by 1. Here we are not merging the tables. The code 5 is having the highest count. So the trending Topics are "OnoPlus One" "India" and "Cyanogen".

Conclusion: Since this model is an ongoing process work of our work first time we are applying LZW for frequent pattern mining. The model will be expanded and modified by us in upcoming days. When we compared it with FP-Growth algorithm's construction of FP-Tree on each core, this new model will reduced the workload and memory consumption of the FP-Trees.

References:

1 Krishna Gadia and Kiran Bhowmick, "Parallel text mining in multi core systems using FP-Tree algorithm", Science Direct Procedia Computer Science 45(2015)111-117.
2 Sharmila Nasreen, Mohammed Azam ..., "Pattern mining Algorithm for finding Associated Frequent Patterns for Data Streams: A Survey", Science Direct Procedia Computer Science,37(2014)109-116.
3 Gurneet Kaur", Associated Rule Mining: A Survey", IJCSIT VOL5(2),2014,2320-2324.
4 Firat Tekiner, Yoshimasa ect..," Parallel Text Mining for Large Text Processing, EMSSN-4 348 CSNDSP 2010.
5 Gonzalez," Digital Image Processing", Boo,. Third edition.
6 Mrs. Sayantani Ghosh, Mr. Sudipta Roy and Prof. Samir K, "A Tutorial Review on Text Mining Algorithms", IJARCCE, ISSN 2278-1027, Vol1,Issue4,June 2012.
7 Jakob Uszkoreit, Jay M Ponte ect..," Large Scale Parallel Document Mining for Machine Translation", Proceedings of Coling 2010, Pages 1101-1109
8 Charu C. Aggarwal, Ghengxiang Zhai," Mining Text Data", Kluwer Academic Publishers, London, Book.
9 Vishal Gupta, Gurpreet S. Lehal, "A Survey of Text Mining Techniques and Applications", Journal of Emerging Technologies in Web Intelligence, VOL., 1, No 1, Agu 2009.

***