Volume 3, Issue 5, May 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Execution of Parallel and Distributed Frequent-Regular Pattern Mining in large databases

Munichandra. Obbu*
M.Tech 2nd year, Dept of CSE,
ASCET, GUDUR, India.
muni.chandra111@gmail.com

Alagala. Bhaskar
Associate Professor, Dept of CSE,
ASCET, GUDUR, India.
alagala.bhaskar@gmail.com

C. Rajendra
Professor & HOD ,Dept of CSE,
ASCET, GUDUR, India.
hod.cse@audisankara.com

Abstract— A good number of parallel and distributed frequent pattern mining algorithms have been proposed so far for the large and/or distributed databases. Not only occurrence frequency of a pattern but also occurrence behaviour (regularity) of a pattern may be treated as an emerging area in data mining research. So far some efforts have been made to mine regular patterns but there is no suitable algorithm exists to mine frequent-regular patterns in parallel and distributed environment. Therefore, in this paper we introduced a new method called PFRP-method (Parallel Frequent Regular Pattern-method) to discover frequent-regular patterns in large databases using vertical data format which requires only one database scan. Our method works in parallel at each local site in order to reduce I/O cost and inter-process communication, generates all frequent-regular patterns in the final phase. Our experiment results show that our PFRP-method is highly efficient in large databases frequent-regular patterns in the final phase. Our experiment results show that our PFRP-method is highly efficient in large databases.

Keywords— frequent patterns, regular patterns, large databases, vertical data format, parallel and distributed algorithms

I. INTRODUCTION

Frequent pattern mining was the fundamental and important unit in data mining research. The Apriori algorithm was the first algorithm to find frequent item sets that was introduced by Agrawal et. al in 1993. This algorithm uses revious knowledge and employs an iterative approach known as level-wise search to generate frequent patterns (FP). First, it cans the database to generate 1-itemsets, recursively it generates 2-itemset and then 3-itemset and continues until all the frequent item sets were generated. It requires k database scans to generate k-itemset and large amount of memory which reduces the efficiency of the algorithm. In the year 2000 Han et.al [3] introduced frequent pattern tree (FP-tree) and FP-growth algorithm to generate frequent patterns without candidate generation. This algorithm has been found to be efficient in memory as well as in execution time when compared to apriori algorithm because it needs only two database scans to find out the complete set of frequent items in a database. An item set is said to be frequent when the occurrence frequency of an item is not less than the user given minimum support threshold.

II. RELATED WORK

In this section, we discuss on regular pattern mining, vertical data format and parallel and distributed frequent pattern mining. Recently, Tanbeer et. al [4] introduced a new problem to mine regular patterns in a database which follows a temporal regularity in the occurrence behavior of a pattern. They proposed a tree-based data structure called RP-tree which captures user-given maximum regularity threshold and mines regular patterns from the database with two database scans. It creates an item header table, called regular table (R-table) to store items with respective regularity and support in the first scan. Then in the second scan the RP-tree is constructed in support descending order only for regular items in R-table. The construction process of an RP-tree is similar to FP-tree construction technique however in RP-tree no node maintains the support count instead it maintains the transaction_ids. Ruggieri [13] introduces Regular Mine, for mining concise representation of frequent itemsets. It computes regular itemsets from the frequent item sets in the class of equivalence of a closed one.

The advantages of using vertical data format [11], [12], [14] are it requires only one database scan, it reduces input/output operations, it uses simple operationslike union, intersection, deletion etc., it judges non regular itemsets before generating candidate itemsets. Therefore, in this paper we used vertical data format (i.e., itemset : tid_set) to mine frequent-regular patterns with PFRP-method. A good number of algorithms have been developed so far to mine frequent patterns in different domains. Zaki [12] conducted a survey on association rule mining and its parallelization schemes. Most of the algorithms are based on apriori and FP-tree in parallel and distributed data mining. Orlando et al. [6] make best use of count distribution, data distribution and candidate distribution. Savesere et al. [7] proposed a two pass partition algorithm which divides horizontal database into vertical partitions, gnerates all local frequent itemsets through tidlist intersection and with second pass it generates all global frequent itemsets. The performances of the Apriori based algorithms are limited with the capabilities of Apriori principle. The algorithms FPForest [8], Load Balancing FP-tree [9]

and Parallel Patter-tree [10] which are based on FP-growth mining distributes large database into several small parts to each of the local sites. After that they construct their local FP-trees separately at each site according to their FP-tree construction procedure and mine for global frequent patterns at the final stage. In this paper we consider [4] [10] to mine global frequentregular patterns in large databases using vertical database format in parallel and distributed environment

PARALLEL AND DISTRIBUTED FREQUENT-REGULAR PATTERN MINING

In this section we describe parallel and distributed frequent-regular pattern mining with our proposed PFRP-method. In this regard we consider a distributed environment containing different locations (sites) where each location contains a processor, memory and other resources. Divide the large database into small, nonoverlapping and equal partitions in order to distribute for n-number of sites. We discuss parallel and distributed frequent-regular pattern mining with our example database DB.

Table 1. Example Database *DB*

S_0		S_1	
Tid	*Transaction*	*Tid*	*Transaction*
1	a, b, c, d	1	b, c, d, e
2	a, b	2	a, c, d
3	a, c, d, e	3	a, d, e
4	b, d, e	4	a, b, c, d, e
5	a, c, d, e	5	a, c, d, e
6	b, c, d	6	c, d
7	a, d, e	7	b, c, d
8	a, b, c, d, e	8	b, d, e
9	a, b, c, e	9	a, b, c

Table 2. Vertical Database

S_0		S_1	
Items	*Tids*	*Items*	*Tids*
a	1, 2, 3, 5, 7, 8, 9	a	2, 3, 4, 5, 9
b	1, 2, 4, 6, 8, 9	b	1, 4, 7, 8, 9
c	1, 3, 5, 6, 8, 9	c	1, 2, 4, 5, 6, 7, 9
d	1, 3, 4, 5, 6, 7, 8	d	1, 2, 3, 4, 5, 6, 7, 8
e	3, 4, 5, 7, 8, 9	e	1, 3, 4, 5, 8

Table 3. PFRP Header Table

Items	S_0	S_1	--	S_n	Total
i_1	$Reg_0(i_1)$	$Reg_1(i_1)$	--	$Reg_n(i_1)$	$Max_i(Reg_i(i_1))$
	$Sup_0(i_1)$	$Sup_1(i_1)$	--	$Sup_n(i_1)$	$\sum_i(Sup_i(i_1))$
i_2	$Reg_0(i_2)$	$Reg_1(i_2)$	--	$Reg_n(i_2)$	$Max_i(Reg_i(i_2))$
	$Sup_0(i_2)$	$Sup_1(i_2)$	--	$Sup_n(i_2)$	$\sum_i(Sup_i(i_2))$
⋮	⋮	⋮	--	⋮	⋮
i_m	$Reg_0(i_m)$	$Reg_1(i_m)$	--	$Reg_n(i_m)$	$Max_i(Reg_i(i_m))$
	$Sup_0(i_m)$	$Sup_1(i_m)$	--	$Sup_n(i_m)$	$\sum_i(Sup_i(i_m))$

Table 1 is the horizontal database that contains two sites S0 and S1having nine transactions each. We require only one database scan to mine frequentregular patterns by using PFRP-method. At each local processor the horizontal database converts into vertical database with one database scan which are shown in Table 2. Let us consider the global maximum regularity of a pattern max_reg, □ = 4 and the global minimum support min_sup, □ = 8 are two measures to mine global frequent-regular patterns with the hep of Table 3 (i.e., PFRP-header table). There is a header table array which is somewhat similar to PP-tree [10] header table that collects all local max_reg's and all min_sup's respectively. After converting the database into vertical format the mining process starts with the proposed PFRP-method by finding all the periods of length-1 items at each of their individual local sites (i.e., P0X, P1X, . . . , PnX). To obtain the periods of a pattern from the above two processors in Table 1, let us consider the first transaction be a null transaction say tfirst = 0 and the last transaction is the ninth transaction say tlast = 9. For example, in site S1 item e occurs in 1, 3, 4, 5 and 8 transactions respectively. The periods computation of e are (1 - tfirst) 1, (3 - 1) 2, (4 - 3) 1, (5 - 4) 1, (8 - 5) 3, (9 - 8) 1, where tfirst = 0 and tlast = 9. Therefore the maximum period of item e in S1 is called a regularity of a pattern i.e., maxe (1, 2, 1, 1, 3, 1) = 3 which can be seen in Table 4. PFRP-header table (Table 5) collects all local supports and local

regularities of length-1 itemset to identify according to the given two thresholds (i.e., □ = 4 and □ = 8). In the process of PFRP-method, it calculates maximum regularity measure among all local regularities (i.e., Reg(X) = Max(Reg1, Reg2, . . . , Regn) ≤ □) for example, in local site S0, P0

Table 4. PFRP-local tables with P_i^X, Sup_i and Reg_i

S_0				S_1			
Items	P_0^X	Reg_0	Sup_0	Items	P_1^X	Reg_1	Sup_1
a	1, 1, 1, 2, 2, 1, 1	2	7	a	2, 1, 1, 1, 4	4	5
b	1, 1, 2, 2, 2, 1	2	6	b	1, 3, 3, 1, 1	3	5
c	1, 2, 2, 1, 2, 1	2	6	c	1, 1, 2, 1, 1, 1, 2	2	7
d	1, 2, 1, 1, 1, 1, 1, 1	2	7	d	1, 1, 1, 1, 1, 1, 1, 1, 1	1	8
e	3, 1, 1, 2, 1, 1	3	6	e	1, 2, 1, 1, 3, 1	3	5

Table 5. PFRP-header table with global regularities and global supports

Items	Reg_0	Reg_1	$Max(Reg_0,Reg_1)$	Sup_0	Sup_1	Sup_0+Sup_1
a	2	4	4	7	5	12
b	2	3	3	6	5	11
c	2	2	2	6	7	13
d	2	1	2	7	8	15
e	3	3	3	6	5	11

maxe (3, 1, 1, 2, 1, 1) = 3 which is less than 4 and in local site S1, P1eis maxe (1, 2, 1, 1, 3, 1) = 3 and Reg(e) finds the global regularity is Max (3, 3) = 3. So, Reg(e) satisfies the regularity threshold (□ = 4) so e is a regular item. And then it finds for global minimum support threshold from all the sum of supports (Sup(X) = Sum(Sup1, Sup2, . . . , Supn) available in the header table. For the same example Sup(e) = 6 + 5 = 11 which also satisfies the support □ = 8. So in this example item e is a frequent regular item. If Reg(e) in S1 did not satisfies the regularity threshold or/and support threshold in PFRP-header table it removes e from the table. Table 5 summarizes all global regularity values and global support values of lenth-1 itemset for our running example. For further processing with simple intersection operation we mine for length-2 itemset ignoring all items tha do not satisfy, since frequent-regular patterns follow the downward closure property. In the running example all length-1 items satisfies the given two thresholds, so all the items will be processed to form length-2 itemset. In Table 6 we can see the frequent-regular patterns of length-2.In Table 6 itemset (a, b) did not satisfied the two given measures, (b, c) did not satisfied regularity measure, (b, e) and (c, e) did not satisfied the support threshold. So all these four length-2 itemsets are not frequent-regular therefore they will be ignored. Again the same process continues until no frequent-regular itemsets found. The advantage of our proposed method is, it takes less time to convert the database into vertical format and the mining process takes place individually at all local sites without any inter-process communication among the processors. At the final phase,

Table 6. PFRP-header table with length-2 global regularities and global supports

Items	Reg_0	Reg_1	$Max(R_0,R_1)$	Sup_0	Sup_1	Sup_1+Sup_2
~~a, b~~	~~6~~	~~5~~	~~6~~	~~4~~	~~2~~	~~6~~
a, c	3	4	4	5	4	9
a, d	2	4	4	5	5	10
a, e	3	4	4	5	3	8
~~b, c~~	~~5~~	~~3~~	~~5~~	4	4	8
b, d	3	3	3	4	4	8
~~b, e~~	4	4	4	~~3~~	~~3~~	~~6~~
c, d	2	2	2	5	6	11
~~c, e~~	3	4	4	~~4~~	~~3~~	~~7~~
d, e	3	3	3	5	5	10

the PFRP-header table collects all global regularities and global supports to find out the itemsets which are frequent-regular. However, our PFRP-method in parallel and distributed environment requires minimum inter-process

communications overhead because during mining process it doesn't requires inter-process communication. The only communication it requires is at the time of constructing header tables. So our proposed method is highly efficient in handling large databases.

III. EXPERIMENT RESULTS

In this section we are going to produce our results. Since there is no existing algorithm to discover frequent-regular patterns, we only examine PFRP-method performance which includes converting database into vertical format and VDRP-table and VDRP-Header table and corresponding parallel mining process. We used our PFRP-method over synthetic (T1014D100K) and real (Kosarak) datasets which are often used for frequent pattern mining experiments developed at IBM Almaden Quest research group and obtain from http://cvs.buu.ac.th/mining/Datasets/synthesis_data/

and UCI Machine Learning Repository (University of California – Irvine, CA) respectively. We consider all local sites with identical configuration which consists of

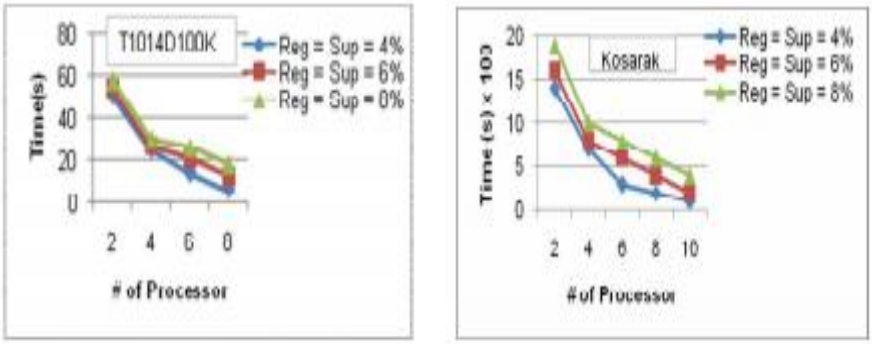

Figure 1. Execution Time over different λ and δ

2.66 GHz CPU with 2GB main memory running on Windows XP. All programs are written in NetBeans IDE 7.0. Message passing interface is established for communication among all the sites. We distributed the partitioned database among the sites and the processor in the corresponding site will have complete access to its portion of the database. We report the results on T1014D100K synthetic dataset which contains 100K transactions, 870 items and 10.10 as the average transaction length. The above Figure 1 shows the execution time on different Reg() and Sup() values on increasing number of processors. We also report on the Kosarak real dataset that contains 990K transactions, 41,270 items and 8.10 as the average transaction length. Figure 2 shows the execution time on various Reg() and Sup() values.

IV. CONCLUSIONS

Parallel computing is a necessary component in any large-scale data mining application. In large databases the performance of the parallel algorithms completely based on I/O cost and inter-process communication. In this paper we have introduced a new mining method called PFRP-method in parallel and distributive environment to obtain the complete set of frequent-regular patterns reducing I/O cost and inter process communication. It requires only one database scan to convert the horizontal database into vertical data format, I/O cost reduced and also no inter-process communication takes place at the mining process between the processors. This shows the efficiency of our proposed method.

REFERENCES

1. Agrawal R., Imielinski., Swami N.: Mining association rules between sets of items in large databases. In ACM SIGMOD Int.Conf.on Management of Data, pp. 207-216 (1993)
2. Agrawal R and Srikanth R.: Fast algorithms for mining association rules. In VLDB, pp.489- 499 (1994)
3. Han J., Pei J., Yin Y.: Mining frequent patterns without candidate generation. In ACM SIGMOD Int. Conf. on Management of Data, pp.1-12 (2000)
4. Tanbeer S K., Farhan C.A., Jeong B-S., Lee Y-K.: Mining regular patterns in transactional database. In IEICE Trans., pp. 2568-2577 (2008)
5. Tanbeer S K., Farhan C.A., Jeong B-S., Lee Y-K.: Discovering periodic-frequent patterns in transactional databases. In PAKDD 2009, pp. 242-253 (2009)
6. Orlando S., Palmerini P., Perego R., Silvestri F.: An efficient parallel and distributed algorithm for counting frequent sets. In VECPAR, pp. 421-435 (2003)
7. Savasere A., Omiecinski E., Navathe S.B.: An efficient algorithm for mining association rules in large databases. InVLDB pp. 432-444 (1995)
8. Hu J. and Yang-Li X. A fast parallel association rules mining algorithm based on FP-Forest. In 5th international symposium on Neural Networks, pp. 40-49 (2008)
9. Yu K.-M., Zhou J., Hsiao W.C.: Load balancing approach parallel algorithm for frequent pattern mining. In PaCT,pp. 623-631 (2007)

10. Tanbeer S K., Farhan C.A., Jeong B-S.: Parallel and distributed algorithms for frequent pattern mining in large databases. In IETE technical review, vol.26, pp. 55-66 (2009)
11. Yi-ming G., and Zhi-jun W.: A vertical format algorithm for mining frequent itemsets. In IEEE transactions, pp.11-13 (2010)
12. Zaki M. J.: Parallel and distributed association mining: A survey. In IEEE concurrency, pp. 14-25 (1999)
13. Ruggieri S.: Frequent Regular Itemset Mining. In: ACM KDD (2010)
14. Zaki M.J., Gouda K.: Fast Vertival Mining using Diffsets. In. ACM SIGKDD'03 (2003)

