Association Rule Mining by Block Scattered Transposition

Gurudatta Verma,Vinti Nanda

Abstract—**Association Rule Mining technique is used to discover the interesting association or correlation among a large set of data items. It plays an important role in generating frequent itemsets from large databases. The Association Rule Mining algorithms such as Apriori, FP-Growth requires repeated scans over the entire database. All the input/output overheads that are being generated during repeated scanning the entire database decrease the performance of CPU, memory and I/O overheads. In this paper, we have proposed An Effectual Generalized Mesh Transposition Algorithm (EGMTA) for frequent itemsets generation. Our algorithm considers transactional dataset as Matrix and Block Scattered Matrix Transposition is applied on generalized transactional dataset. The CPU and I/O overhead can be reduced in our proposed algorithm and it is much faster than other Association Rule Mining algorithms**.

***Keywords*-Data Mining, Association Rule Mining (ARM), Association rules, Apriori algorithm, Frequent pattern**.

I. Introduction

The rapid development of computer technology, especially increased capacities and decreased costs of storage media, has led businesses to store huge amounts of external and internal information in large databases at low cost. Mining useful information and helpful knowledge from these large databases has thus evolved into an important research area [3, 2, 1]. Data mining commonly involves four classes of task:

Classification - Arranges the data into predefined groups. For example an email program might attempt to classify an email as legitimate or spam. Common algorithms include nearest neighbor, Naive Bayes classifier and neural network.

Clustering - Is like classification but the groups are not predefined, so the algorithm will try to group similar items together.

Regression - Attempts to find a function which models the data with the least error. A common method is to use Genetic Programming.

Association rule learning - Searches for relationships between variables. For example a supermarket might gather data of what each customer buys. Using association rule learning, the supermarket can work out what products are frequently bought together, which is useful for marketing purposes. This is sometimes referred to as "market basket analysis".

Association Rule Mining (ARM) algorithms are defined into two categories; namely, algorithms respectively with candidate generation and algorithms without candidate generation. In the first category, those algorithms which are similar to Apriori algorithm for candidate generation are considered. In the second category, the FP-Growth algorithm is the best–known algorithm.

The main drawback of apriori algorithm is the repeated scans of large database. This may be a cause of decrement in CPU performance, memory and increment in I/O overheads. The performance and efficiency of ARM algorithms mainly depend on three factors; namely candidate sets generated, data structure used and details of implementations [8].

Performance Based Transposition Algorithm[17] uses these three factors. Transactional database is considered as a two dimension array which works on generalized value dataset. The main difference between proposed algorithm and other algorithms is that instead of using transactional array in its natural form, our algorithm uses transpose of array i.e. rows and columns of array are interchanged and transposition using parallel matrix transpose algorithm. The advantage of using transposed array is to calculate support count for particular item. There is no need to repeatedly scan array. Only by finding the row sum of the array will give the required support count for particular item, which ultimately results in increased efficiency of the algorithm. In the first pass of algorithm, we will receive all the support count value for the 1-itemset. Mining of association rules is a field of data mining that has received a lot of attention in recent years.

Following table defines the comparison among these algorithms.

Algorithm	Dataset	CPU overhead	Preprocessing on Dataset
Apriori	Boolean	Less	Less than PBTA
PBTA	Boolean	More	More

While the work is pioneering, PBTA approach works only for transposed boolean dataset. So it requires separate application for generalized dataset to Boolean dataset conversion as well as separate application for dataset transposition. We are making an integrated application which take generalized dataset as input and uses parallel mesh transposition method for dataset transposition.

The remainder of this paper is organized as follows: Section 2 provides a brief review of the related work. In Section 3, we explain Frequent Itemset and Association Rule Mining through Apriori Algorithm. In Section 4, we introduce our proposed EGMTA algorithm. Proposed methodology of the algorithm and comparative analysis is presented in section 5 and section 6 respectively. Finally, we concluded our work.

II. RELATED WORK

One of the most well known and popular data mining techniques is the Association rules or frequent item sets mining algorithm. The algorithm was originally proposed by Agrawal et al. [4] [5] for market basket analysis. Because of its significant applicability, many revised algorithms have been introduced since then, and Association rule mining is still a widely researched area.

Agrawal et. al. presented an AIS algorithm in [4] which generates candidate item sets on-the-fly during each pass of the database scan. Large item sets from previous pass are checked if they are present in the current transaction. Thus new item sets are formed by extending existing item sets. This algorithm turns out to be ineffective because it generates too many candidate item sets. It requires more space and at the same time this algorithm requires too many passes over the whole database and also it generates rules with one consequent item.

Agrawal et. al. [5] developed various versions of Apriori algorithm such as Apriori, AprioriTid, and AprioriHybrid. Apriori and AprioriTid generate item sets using the large item sets found in the previous pass, without considering the transactions. AprioriTid improves Apriori by using the database at the first pass. Counting in subsequent passes is done using encodings created in the first pass, which is much smaller than the database. This leads to a dramatic performance improvement of three times faster than AIS.

Scalability is another important area of data mining because of its huge size. Hence, algorithms must be able to “scale up” to handle large amount of data. Eui-Hong et. al [16] tried to make data distribution and candidate distribution scalable by Intelligent Data Distribution (IDD) algorithm and Hybrid Distribution (HD) algorithm respectively. IDD addresses the issues of communication overhead and redundant computation by using aggregate memory to partition candidates and move data efficiently. HD improves over IDD by dynamically partitioning the candidate set to maintain good load balance. Different works are reported in the literature to modify the Apriori logic so as to improve the efficiency of generating rules. These methods even though focused on reducing time and space, in real time still needs improvement.

III. FREQUENT ITEM SET AND ASSOCIATION RULE

The aim of Association rule mining is exploring relations and important rules in large datasets. A dataset is considered as a sequence of entries consisting of attribute values also known as items. A set of such item sets is called an item set. Frequent item sets are sets of pages which are visited frequently together in a single server session. Only the list of session IDs and URLs is used during this process. Support is often utilized to limit the number of discovered patterns. [17] Support of the subset {i1… in} from a set D is defined as in equation (1).

$$S\ (i_1,\ i_n) = \text{count}\ (\{i_1,\ i_n\} \sqsubseteq D)\ /\ \text{Count}(D)\text{----}\ (1)$$

Once the frequent item sets are discovered, we calculate for each item set the interest to objectively rank them. Interest is defined as in equation (2).

$$I\ (i_1,\ldots,\ i_n) = \quad S(i_1,\ldots,\ i_n)\ \ /\ \prod_{j=1}^{n} S(ij)\text{--------}(2)$$

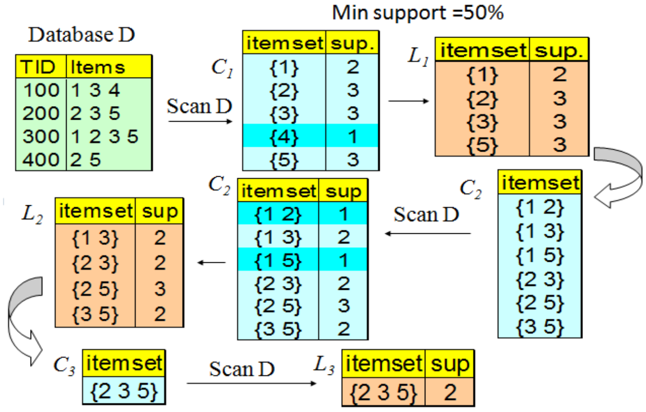
Set of n frequent items are broken into n separate Association rules. The confidence of an association rule (as in equation (3)) is the fraction of sessions where the subsequent and the antecedent are present and sessions where only the subsequent is present.

For the rule ia →is1⋯ isn it is

$$C(i_a \rightarrow is_1,\ldots,\ is_n) = S(i_a \rightarrow is_1,\ldots,\ is_n)\ /\ S(i_a)\ \text{---}(3)$$

A. Apriori Algorithmt

ARM is one of the promising techniques of data mining to extract interesting correlations, frequent patterns, associations or casual structures among sets of items in the transaction databases or other data repositories. The advantage of the algorithm is that before reading the database at every level, it prunes many of the sets which are unlikely to be frequent sets by using the Apriori property, which states that all nonempty subsets of frequent sets must also be frequent. This property belongs to a special category of properties called anti-monotone in the sense that if a set cannot pass a test, all of its supersets will fail the same test as well. Using the downward closure property and the Apriori property the algorithm works as follows. The first pass of the algorithm counts the number of single item occurrences to determine the L1 or single member frequent itemsets. Each subsequent pass, K, consists of two phases. First, the frequent itemsets Lk-1 found in the (k-1)th pass are used to generate the candidate itemsets Ck, using the Apriori candidate generation algorithm. Therefore, the database is scanned and the support of the candidates in Ck is determined to ensure that Ck itemsets are frequent itemsets [10].



TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

itemset	sup
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

itemset	sup
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

itemset
{2 3 5}

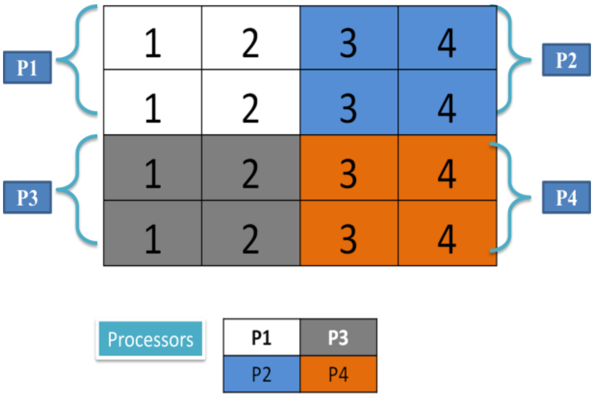
itemset	sup
{2 3 5}	2

IV. Effectual Generalized Mesh Transposition Algorithm (EGMTA)

In Apriori algorithm, discovery of association rules require repeated passes over the entire database to determine the commonly occurring set of data items. Therefore, if the size of disk and database is large, then the rate of input/output (I/O) overhead to scan the entire database may be very high. We have proposed Effectual Generalized Mesh Transposition Algorithm (EGMTA), which improves the Apriori algorithm for repeated scanning of large databases for frequent itemsets generation. In EGMTA, transaction dataset will be used in the transposed form (Transposition is done using Parallel Transposition algorithm) and the description of proposed algorithm is discussed in the following sub-sections.

A. *Transaction dataset transposition*

The idea of our algorithm is quite simple. Matrix is divided into sub matrices according to number of processor threads and sub matrices are distributed among processors as shown in Fig-1.



1	2	3	4
1	2	3	4
1	2	3	4
1	2	3	4

Figure 1. Parallel Matrix transpose.

B. *Matrix Transpose Algorithm*

```
Procedure TRANSPOSE (A)
  Step 1: do steps 1.1 and 1.2 in parallel
        (1.1) for i = 2 to n do in parallel
              for j = I to i - 1 do in parallel
                    C(i- 1, j) (a, j, i)
              end for
        end for
        (1.2) for i = I to n -I do in parallel
              for j = i + I to n do in parallel
                    B(i, rj -1) (ai, j, i)
        end for
        end for
  Step 2: do steps 2.1, 2.2, and 2.3 in parallel
      (2.1) for i = 2 to n do in parallel
            for j = I to i - I do in parallel
            while P(i, j) receives input from its
            neighbors do
            (i) if (aki, m, k) is received from P(i + 1, j)
                  then send it to P(i - 1, j)
             end if
            (ii) if (ak,, m, k) is received from P(i - 1, j)
                  then if i = m andj = k
                  then A(i, j) - a,, {ak, has reached
                  its destination}
                  else send (aki, m, k) to P(i + 1, j)
                  end if
            end if
            end while
            end for
        end for
      (2.2) for i = 1 to n do in parallel
            while P(i, i) receives input from its
            neighbors do
            (i) if (ak., m, k) is received from P(i + 1, i)
                  then send it to P(i, i + 1)
            end if
            (ii) if (ak., m, k) is received from P(i, i + 1)
                  then send it to P(i + 1, i)
            end if
            end while
         end for
      (2.3) for i = 1 to n - 1 do in parallel
            for j = i + I to n do in parallel
            while P(i, j) receives input from its
            neighbors do
            (i) if (ak., m, k) is received from P(i, j + 1)
                  then send it to P(i, j - 1)
              end if
            (ii) if (akin, m, k) is received from P(i, j -1)
                  then if i = m andj = k
                  then A(i, j) +- ak. {ak, has reached
                  its destination}
                  else send (aki, m, k) to P(i, j + 1)
                  end if
```

```
                    end if
                end while
                end for
        end for
```

C. Candidate Generation Algorithm

In the candidate generation algorithm, the frequent itemsets are discovered in k-1 passes. If k is the pass number, Lk-1 is the set of all frequent (k-1) itemsets. Ck is the set of candidate sets of pass k and c denotes the candidate set. l1,l2 …lk are the itemsets[19]. The candidate generation procedure is as follows:

```
Procedure Gen_candidate_itemsets (Lk-1)
    Ck = Φ
    for all itemsets $I_1 \in L_{k-1}$ do
    for all itemsets $l_2 \in L_{k-1}$ do
    if $I_1[1] = I_2[1] \wedge I_1[2] = I_2[2] \wedge \ldots \wedge I_1[k-1] < I_2[k-1]$
then
        c = $I_1[1], I_1[2] \ldots I_1[k-1], I_2[k-1]$
    Ck = $C_k \cup \{c\}$
End Procedure
```

D. Pruning Algorithm

The pruning step eliminates some candidate sets which are not found to be frequent.

```
Procedure Prune(Ck)
    for all c $\in$ Ck
    for all (k-1)-subsets d of c do
    if d $\notin$ Lk-1
    then Ck = Ck – {c}
End Procedure
```

E. EGMTA Algorithm Description

The EGMTA uses candidate generation and pruning algorithms at every iteration. It moves from level 1 to level k or until no candidate set remains after pruning. The step-by-step procedure of EGMTA algorithm is described as follows.

```
Procedure EPTA()
// Transpose the transactional database
        1. Transpose(Data Set)
        2. Read the database to count the support of C1 to
determine L1 using sum of rows.
        3. $L_1$= Frequent 1- itemsets and k:= 2
        4. While (k-1 $\neq$ NULL set) do
        Begin
        $C_k$: = Call Gen_candidate_itemsets ($L_k$-1)
        Call Prune ($C_k$)
        for all itemsets i $\in$ I do
        Calculate the support values using dot-multiplication
of array;
        $L_k$ := All candidates in Ck with a minimum support;
        k:=k+1
        End
        5. End of step-4
End Procedure
```

V. Proposed Methodology

Suppose we have a transactional database in which the user transactions from T1 to T5 and items from A1 to A5 are stored in the form of generalized values, which is shown in Table 1. Our algorithm uses transpose of array i.e. rows and columns of array are interchanged. The advantage of using transposed array is to calculate support count for particular item. There is no need to repeatedly scan array. Only by finding the row sum of the array will give the required support count for particular item, which ultimately results in increased efficiency of the algorithm.

VI. Comparative Analysis

VII. Conclusions

We have described an algorithm which is not only efficient but also fast for discovering association rules in large databases. An important, contribution of our approach is that it drastically reduces the I/O overhead associated with Apriori algorithm. This algorithm may prove useful for many real-life database mining scenarios where the data is stored in generalized form. Currently this algorithm uses Mesh parallel transposition algorithm so still preprocessing overhead (transposition) need to improve.

References

[1] C.-Y. Wang, T.-P. Hong and S.–S. Tseng. "*Maintenance of discovered sequential patterns for record deletion"*. Intell. Data Anal. pp. 399-410, February 2002.

[2] M.S. Chen, J.Han and P.S. Yu. "*Data Mining : An overview from a database perspective*", IEE Transactions on Knowledge and Data Engineering 1996.

[3] R.Agrawal, T. Imielinksi and A. Swami, "*Database Mining: a performance perspective*", IEE Transactions on knowledge and Data Engineering, 1993.

[4] Agrawal, R., Imielinski, T., and Swami, A. N. "*Mining Association Rules Between Sets of Items in Large Databases*". Proceedings of the ACM SIGMOD, International Conference on Management of Data, pp.207- 216, 1993.

[5] Agrawal. R., and Srikant. R., "*Fast Algorithms for Mining Association Rules*", Proceedings of 20th International Conference of Very Large Data Bases. pp.487-499,1994.

[6] Jong Park, S., Ming-Syan, Chen, and Yu, P. S. "*Using a Hash-Based Method with transaction Trimming for Mining Association Rules"*. IEEE Transactions on Knowledge and Data Engineering, 9(5), pp.813-825,1997.

[7] M.H.Margahny and A.A.Mitwaly, "*Fast Algorithm for Mining Association Rules*" in the conference proceedings of AIML, CICC, pp(36-40) Cairo, Egypt, 19-21 December 2005.

[8] Y.Fu., "*Discovery of multiple-level rules from large databases*", 1996.

[9] F.Bodon, "*A Fast Apriori Implementation*", in the Proc.1st IEEE ICDM Workshop on Frequentc Itemset Mining Implementations (FIMI2003, Melbourne,FL).CEUR Workshop Proceedings 90, A acheme, Germany 2003.

[10] Akhilesh Tiwari, Rajendra K. Gupta, and Dev Prakash Agrawal, "*Cluster Based Partition Approach for Mining Frequent Itemsets*" in the International Journal of Computer Science and Network Security(IJCSNS), VOL.9 No.6,pp(191-199) June 2009.

[11] Frequent Itemset Mining Implementations (FIMI'03) Workshop website, http://fimi.cs.helsinki.fi, 2004.

[12] M.J. Zaki. "*Scalable algorithms for association mining*". IEEE Transactions on Knowledge and Data Engineering, 12 : 372 –390, 2000.

[13] Jochen Hipp, Ulrich G¨untzer, Gholamreza Nakhaeizadeh. "*Algorithms for Association Rule Mining – A General Survey and Comparison*".ACM SIGKDD, July 2000, Vol-2, Issue 1, page 58-64.

[14] Sotiris Kotsiantis, Dimitris Kanellopoulos. "*Association Rules Mining: A Recent Overview*". GESTS International Transactions on Computer Science and Engineering, Vol.32 (1), 2006, pp. 71-82.

[15] S. Brin, R. Motwani, J. D. Ullman, AND S. Tsur, "*Dynamic itemset counting and implication rules for market basket data"*, SIGMOD Record 26(2), pp. 255–276, 1997.Kim Man Lui, Keith C.C. Chan, and John Teofil Nosek "The Effect of Pairs in Program Design Tasks" IEEE transactions on software engineering, VOL. 34, NO. 2, march/april 2008.

[16] Eui-Hong Han, George Karypis, and Kumar, V. "*Scalable Parallel Data Mining for Association Rules*" IEEE Transaction on Knowledge and Data Engineering, 12(3), pp.728-737, 2000.

[17] Sanjeev Kumar Sharma & Ugrasen Suman "*A Performance Based Transposition Algorithm for Frequent Itemsets Generation*" International Journal of Data Engineering (IJDE), Volume (2) : Issue (2) : 2011

[18] Gurudatta Verma & Vinti Nanda "*An Effectual Algorithm For Frequent Itemset Generation In Generalized Data Set*" Using Parallel Mesh Transposition"Proceedings of EEE International Conference on Advances in Engineering, Science and Management,2012

Gurudatta Verma, Department of Computer Science & Engineering, Chhatrapati Shivaji Institute of Technology, Durg, India, e-mail: gurudatta.verma@gamil.com.

Vinti Nanda, Department of Computer Science & Engineering, Chhatrapati Shivaji Institute of Technology, Durg, India, e-mail: vintinanda@csitdurg.in