Abstract
In frequent itemset mining problem, there is a necessity to handle different data types and mining applications. In order to improve the efficiency in mining frequent itemset, a novel itemset representation is proposed called DiffNodeset. Based on DiffNodeset, dFIN Algorithm is introduced for mining frequent itemset. dFIN Algorithm finds frequent itemset using set enumeration tree and superset equivalence property. Hence the dFIN algorithm minimizes the running time and memory consumption when compared to existing leading algorithms. The advantage of DiffNodeset lies in that its size is much smaller. This makes DiffNodeset more suitable for mining frequent itemsets. With frequent itemset it is also possible to identify infrequent items that have support less than the threshold. By associating an infrequent item with a frequent itemset, the proposed work improves the sales of infrequent items. The association is based on the expiry date of infrequent itemset and support count of frequent itemset.

I. INTRODUCTION
Data mining is the task of examining data from different views and transforming it into useful facts and information. The information can be used to increase income, cuts costs, or sales surplus. Data mining software is one of a number of analytical tools for analyzing
data[15]. It allows users to inspect data from many different dimensions, analyse it, and summarize the relationships identified. Data mining is the process of finding correlations, patterns or relationships among millions of fields in large relational databases.

Data mining is mainly used today by companies with a strong consumer attention - retail, financial, communication, and marketing organizations. It give assistance to these companies to determine relationships among "internal" factors such as cost, product positioning, or staff skills, and "external" factors such as economic indicators, and customer demographics. And it enables them to determine the impact on sales, customer satisfaction, and corporate profits. Finally, it helps them to access information to view detail transactional data. Frequent itemset mining is a branch in data mining that focus on sequences of actions or events. The common task of frequent itemset mining is to find the frequent items in a particular transaction.

Frequent itemset refers to items which are frequently purchased. More specifically, it finds set of products that are frequently bought together. For example, if a customer buys table and chair, then she/he will probably also buy table cloth. Identifying frequent itemset is one of the most important issues faced by the data mining community.

The support supp(X) of an itemset X is defined as the indication of number of transactions in the dataset which contain the itemset.

\[
\text{supp}(X) = \frac{\text{no. of transactions which holds the itemset } X}{\text{total no. of transactions}}.
\]

There are many structures discovered to mine the frequent itemset. They are efficient to mine the itemset. Nodesets require only the pre-order or post order of each node, which makes it saves half of memory compared with N-lists and Node-lists[2]. Given a set of items \( I = \{i_1, i_2, \ldots, i_n\} \) where \( i_1, i_2, \ldots, i_n \), a set-enumeration tree can be constructed as follows. Initially, the creation of the root node starts and the \( n \) child nodes of the root representing \( n \) 1-itemsets are created. The Node-list is more compact when compared with vertical structure (tidset or diffsets) because transactions with common prefix share the same nodes of the PPC-tree. Second, the support count is transformed into intersection of Node-lists and the complexity of intersecting two Node-lists can be reduced to \( O(m+n) \) by an efficient strategy, where \( m \) and \( n \) are the cardinalities of the two Node-lists respectively.

In this paper, we propose an efficient structure called DiffNodeset to mine the items which are frequently purchased. The items are retrieved based on the dFIN Algorithm. The dFIN Algorithm works based on the set enumeration tree and superset equivalence property. The infrequent items are retrieved based on the minimum threshold. The sales efficiency of infrequent items are performed by associating an infrequent item with least expiry date and the largest frequent pattern.

II. LITERATURE SURVEY

The itemsets are mined using DiffNodeset structure. An algorithm called dFIN is proposed in order to mine the itemsets efficiently(Zhi-Hong Deng 2016)[11]. Initially the PPC tree is constructed to mine F1 itemset. The database is scanned to mine all 1-itemset with support count. All the infrequent items are removed based on the minimum support count. Based on the DiffNodeset structure 2-itemset are mined by ancestor descendant relationship. Finally k-
Continuous 42nd Edition, Page No: 6451-6457

A. Lavanya , S. Rajalakshmi:: An Associated mining approach for promoting sales of infrequent items using AIF Algorithm

itemset are mined using set enumeration tree. All possible pattern can be observed using set enumeration tree. A vertical algorithm called PPV is proposed for fast frequent pattern discovery. PPV obtains Node-lists of each frequent itemset. (Z.H.Deng , Z.H.Wang 2010)[4]. Then PPV obtains Node-lists of the candidate patterns of length k and discovers the frequent patterns of length (K+1). It transforms the frequent patterns into intersecting of Nodelist which makes mining process simple and adopts an efficient method for intersecting two Node-lists which has an average time complexity of O(m+n).

An efficient data structure called nodeset is proposed. Nodeset requires only preorder which consumes half of the memory when compared with N-list and Node list. Based on the Nodeset structure an efficient algorithm called FIN is proposed for mining frequent itemset efficiently (Zhi-Hong Deng, Sheng-Long Lv (2014))[2]. FIN adopts promotion which is based on superset equivalence property as pruning strategy.

Prepost+, a high performance algorithm is introduced for mining frequent itemset. It employs N-list to represent itemset and discovers frequent itemset using set enumeration search tree. Especially it employs an efficient pruning strategy named Children-Parent Equivalence pruning to greatly reduce the search space. (Zhi-Hong Deng, Sheng-Long Lv 2015)[3]. This work of Prepost+ is same as that of Prepost. Mining erasable itemset using NC-sets is proposed, which keeps track of complete information used for mining erasable itemsets efficiently. (Zhi Hong Deng, Xiao Ran Xu 2012)[6]. The efficiency of MERIT is achieved with three techniques. First the NC_set is a compact structure which prunes irrelevant data automatically. Second the computation of an itemset is transformed into the combination of NC_sets which can be completed in linear time complexity.

An algorithm is introduced for mining frequent itemsets is presented in a stream of transactions within a limited time horizon. (Luigi Troiano, Giacomo Scibelli (2014))[13]. The proposed algorithm makes use of a test window discard non-frequent itemsets from a set of candidates. The efficiency of this approach depends on the property that the higher the support threshold is, the smaller the test window is. In addition to considering a sharp horizon, a smooth window is considered. Smoothness is determined in both qualitative and quantitative terms. The Window Itemset Shift (WIS) as an alternative solution, which retains a memory of flowing candidates within a reduced test window. This work, the problem of mining frequent itemsets in a stream of transactions is within a limited window. In addition, WIS does not require a pass through the dataset to compute the support.

Processing incremental databases in the itemset mining is important because a huge amount of data has been accumulated continuously in a variety of application fields and users want to obtain mining results from incremental data in efficient way. One of the major problems in incremental itemset mining is that the mining results are very large-scale according to threshold settings and data volumes. Moreover, it is hard to analyze information. Furthermore, not all of the mining results become significant information. In this work, to solve these difficulties, an algorithm is proposed for mining weighted maximal frequent itemset from incremental databases. (Unil Yun, Gangin Lee (2016))[14]. By scanning a given incremental database once, the proposed algorithm can not only conduct its
mining operations suitable for the incremental environment but also it extracts a smaller number of important itemsets compared to previous approaches.

Two novel approaches are proposed to drive the IWI mining process. Two algorithms are proposed that perform IWI and Minimal IWI mining effectively, driven by the proposed measures, are presented.(Luca Cagliero and Paolo Garza (2014))[7]. Given a weighted transaction data set and a maximum IWI-support threshold the Infrequent Weighted Itemset Miner algorithm finds all IWIs whose IWI-support satisfies minimum support threshold. IWI Miner discovers infrequent weighted itemsets instead of frequent (unweighted) ones.

The infrequent weighted itemset are item sets whose frequency of existence in the analyzed data is less than or equal to a maximum threshold. Two algorithms are inspected to find rare itemset, that are infrequent weighted itemset (IWI) and Minimal Infrequent Weighted Itemset (MIWI) and is based on the frequent pattern-growth paradigm. IWI Miner is a FP-growth-like mining algorithm that performs projection-based item set mining.(Nandhini S,Yogesh M and Gunasekaran S. (2015))[9]. FP-growth mining steps are FP-tree creation then Recursive item set mining from the FP tree index and IWI Miner finds infrequent weighted item sets instead of frequent (unweighted) ones.

III. PROPOSED SYSTEM

The proposed framework for frequent itemset generation is based on the DiffNodeset structure. The transaction database is scanned to generate the frequent items. The items are retrieved based on the support threshold. According to the threshold value the itemsets from f1 to fk are generated and mined. The 1-itemsets (F1 itemset) are generated and sorted in support descending order. The Nodeset of frequent items is constructed in the form of preorder, postorder and count. The DiffNodeset of 2 itemset are sorted in preorder ascendant order. Hence f2 items are generated and the count of 2itemset are found. Constructing pattern tree discovers all the f_k itemsets. It depends on the set enumeration tree and all the frequent k-itemsets(k>=3) are generated and the support count is calculated. All the infrequent items are extracted which has the support less than the threshold. The sales of infrequent items are promoted by associating an infrequent item which has got least expiry date with the largest frequent pattern. Hence the sales efficiency of infrequent items is
increased. The Frequent itemset generation is proposed as the design and is shown in Figure 1.

**A. ppc tree construction:**

**ppc tree (preorder postorder codes)**

In the Construction of PPC tree, the input transaction dataset is scanned to find 1-itemset. All the F1-items are retrieved based on the given minimum support threshold (ε). The F1-items are sorted in support descending order. All the infrequent items are deleted and the sorted frequent items are placed in the PPC tree. The PPC tree is scanned to generate the preorder and postorder codes by the preorder traversal.

**B. Build_2_itemset:**

In Build_2_itemset, the nodeset of two itemset are compared. The nodeset comprises of preorder code, postorder code, count of each item. The DiffNodeset of 2-itemset i1i2, denoted as DiffNodesetsi1i2. DiffNodesetsi1i2 = {(x.pre-order, x.count)|x ∈ Nodesetsi1∧(∃y ∈ Nodesetsi2, the node respect to y is an ancestor of the node co responding to x)}.

where Nodesetsi1 and Nodesetsi2 are the Nodesets of items i1 and i2 respectively.

In addition, the elements in DiffNodesetsi1i2 are sorted in pre-order ascendant order.

The non-ancestor nodes are taken as DiffNodeset. Therefore the support of 2-itemset is calculated as follows:

The support of i1i2, support(i1i2), is equal to

\[ \text{support}(i_1i_2) = \text{support}(i_1) - \sum (E \in \text{DN}_{12}) E.\text{count} \]

Equation 3.1 shows the support of 2-itemset generation. The DiffNodeset of 2-itemset is calculated and subtracted from the support of first item.

**C. construction of pattern tree:**

In construction of pattern tree, the k-itemset (k≥3) are generated. It employs set enumeration tree and superset equivalence property. It generates k-itemset (k≥3) extended from frequent 2-itemset. The support is also calculated for all itemset. superset equivalence property is employed to prune the search space. all the possible pattern of the frequent items can be observed using set enumeration tree.

**D. AIF algorithm:**

With Frequent itemset, it is also possible to identify infrequent items that have support less than threshold. By associating an infrequent item with a frequent itemset, the proposed work improves the sales of infrequent item. The association is based on the expiry date of infrequent itemset and support count of frequent itemset.

**INPUT**  : Infrequent items, Database with expiry date D, minimum support threshold(ε)

**OUTPUT** : Association of infrequent item with Frequent itemset.

Start
Scan database with items and expiry date
Set ε = minimum threshold value
Infrequent items are retrieved
For all infrequent items in database D do
If expiry date of infrequent item has least value then
   Map infrequent item with Large frequent pattern(K)
Else
   Map infrequent item with Large frequent pattern(k-1)
End if
End for
End

The above algorithm is made run for every infrequent item with least expiry date. If the item is not sold then it is associated with other large pattern. Hence the sales rate of infrequent items will be increased.

IV. EXPERIMENTAL STUDY

For the frequent itemset generation we collected dataset from http://fimi.uc.ac.be/src and http://www.adrem.ua.ac.be/goethals/software respectively [11]. It contains the dataset like chess, pumsb, kosarak, mushroom and T10I4D100K. In order to evaluate the performance of dFIN it is checked out with all possible datasets. The performance of dFIN is observed with the dataset. The infrequent items are extracted from the dataset and the sales efficiency of the infrequent items are promoted by associating the infrequent item which has got least expiry date with the greatest support count. dFIN Algorithm works best when compared to existing leading algorithms. It consumes less memory and running time. Hence Frequent items are generated using dFIN algorithm and the sales efficiency of infrequent items are increased using AIF Algorithm.

V. CONCLUSION

The frequent itemsets are mined efficiently using DiffNodeset structure. Based on the DiffNodeset structure, dFIN algorithm is introduced for generating frequent itemset efficiently. dFIN finds frequent itemset using set enumeration tree and superset equivalence property. The running time and memory consumption is comparatively reduced with existing leading algorithms. Based on the minimum support threshold value, the infrequent items are pruned and frequent items are generated. With frequent itemset it is possible to extract infrequent items that have support less than threshold value. By associating an infrequent item with a frequent itemset, the proposed work improves the sales of infrequent items. The association is based on the expiry date of infrequent itemset and support count of frequent itemset.

VI. REFERENCES


TO CITE THIS PAPER