International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887
Volume 7 Issue III, Mar 2019- Available at www.ijraset.com

Frequent Itemset Generation for Analyzing Customer Buying Nature using Bit Vector Mining

G. Harini[1], R. Chandradivya[2], P. Monica[3], D. Sudha[4], M. Krishnamurthy[5]
[1, 2, 3] *Student, Department of Computer Science and Engineering, KCG College of Technology, Chennai, India*
[4, 5] *Faculty, Department of Computer Science and Engineering, KCG College of Technology, Chennai, India*

Abstract: ***The paper introduces an efficient algorithm using bit vector to find frequent itemsets from a huge set of itemsets. The existing frequent itemset generation algorithms lack efficiency in terms of time and space. As a result, the need for new frequent itemset mining algorithms that could tackle the new trends are required.***

Here, the purchased items in a transaction are represented in terms of bit vectors. A bit vector is a vector in which each element is represented in terms of bits (so its value is either 0 or 1). The database is scanned only once. The k-frequent itemsets are obtained by performing LOGICAL AND operation between the items in the scanned table. By doing so, the time and space consumed in determining the frequent itemsets is reduced.

Keywords**: *Bit vectors, Frequent itemset mining, Knowledge discovery***

I. INTRODUCTION

Nowadays, supermarkets and e-shopping sites use data mining algorithms to predict the user's buying nature based upon the frequently purchased products. This data can be used to predict the customer's next purchase and suggest those items to the customers based upon the frequency of purchase of items.

As the amount of data keeps increasing, the efficiency of the existing frequent itemset mining algorithms have decreased. To overcome this issue, the proposed system is introduced for determining the frequent itemsets in a particular transaction with reduced time and space consumption.

Data mining is the key advance in Knowledge Discovery process. It is progressively getting to be essential apparatus in separating intriguing learning from extensive databases.

Also, numerous data mining issues include transient perspectives, with precedents extending from designing to logical research, money and medication. Worldly temporal mining is an augmentation of data mining which manages transient information. In this paper, it is discussed about fleeting database for discovering regular things.

An assortment of calculations are as of now executed for mining frequent itemsets. The Apriori and FP-growth calculation are the two most prominent ones. Specifically, Apriori calculation is a breadth-first search algorithm.

Conversely, FP-growth calculation is a depth-first search algorithm, with no candidate generation. While FP-growth just performs two database scans, which makes FP-growth calculation with a higher order magnitude than Apriori. The highlights of FP- growth rouse to structure a differentially private FIM algorithm dependent on the FP-growth calculation. Amid this examination, a down to earth differentially private FIM increases high information utility, a high level of security and high time productivity. It has been appeared utility protection can be improved by decreasing the length of transactions. Existing work demonstrates an Apriori based private FIM algorithm.

It diminishes the length of transactions by truncating transactions (It implies in the transaction that it has more items than the limitations, at that point eliminate items until its length is under the limit In each database scan, to preserve more frequent items, it influences discovered frequent itemsets to re-truncate the transactions Nonetheless, FP-growth just performs two database scans. Because of this it is unimaginable to re-truncate transactions amid the data mining process. Along these lines, the transaction truncating (TT) approach proposed isn't reasonable for FP-growth. In addition, to avoid privacy breach, noise is added to the support of data itemsets. Given a data itemset in X to satisfy differential privacy, the amount of noise added to the support of i data itemset X depends on the number of support computations of i-itemsets. Unlike Apriori, FP-growth is a depth-first search algorithm. It is hard to obtain the perfect number of support computations of I-itemsets during the mining process of a transaction. A native approach for computing the noisy support of I^{th} item is to use the number of all possible i^{th} item. However, it will definitely produce invalid results.

II. LITERATURE SURVEY

There are numerous works in literature that examine about Association rules, Temporal Mining and Frequent Itemsets.

In improved Apriori calculation, the mining productivity is exceptionally unsuitable when memory for database is considered. Distinctive systems were proposed after Apriori as in FP-development [13], which beats all hopeful ages yet have issues on account of no common prefixes inside the data items. Transient FP tree utilizes separate and vanquish procedure for development and crossing of tree which is utilized to decay the mining undertaking into a lot of littler errand which lessens the pursuit space.

However, Temporal FP Tree system is better just when the information is thick Paper [12] is valuable for the retailer to make its own procedure according to the prerequisite of time. The paper presents three techniques: 3D linked array-based strategy, connected tree technique, and average probability-based setup. The objective here is to minimize computational cost by traversing the database only once. The current proposal addresses attribute uncertainty as well as the tuple uncertainty to map large uncertain databases to the proposed data structures. This work also presents algorithms to extract frequent itemsets. The advantage of this work is that it eliminates the additional space issue and provides better execution time.

The paper [15] describes the basic principle of Apriori algorithm and further proposed an improved algorithm that needs to scan the database only once by adding the concepts of transactional weights and by generating a transactional Boolean matrix. The original database data is scanned and stored in the transaction binary Boolean matrix, after which all operations are performed on the matrix. Row reduction is performed on the matrix and weighted transaction, so the computational complexity can be reduced in case of large transaction volume. The proposed algorithm runs faster than the original algorithm when the transaction volume is gradually increased.

A novel algorithm [3] is introduced for mining FT frequent patterns using pattern growth approach. This algorithm stores the original transactional dataset in a highly condensed, much smaller data structure called FT-FP-tree, and the FT-pattern support and item support of all the FT patterns are counting directly from the FT-FP-tree, without scanning the original dataset multiple times. While costly candidate set generations are avoided by generating conditional patterns from FT-FP-tree.

The partition algorithm [8], to further improve the efficiency, it does so by reducing the number of database scans, however, considerable time is still wasted in scanning infrequent candidate itemsets. In this paper [4] considers frequent itemsets mining in transactional databases. They introduced a new accurate single scan approach for frequent itemset mining (SSFIM), a heuristic as an alternative approach (EA-SSFIM), as well as a parallel implementation on Hadoop clusters (MR-SSFIM). The main advantage of this approach is that it generates a fixed number of candidate itemsets independently from the value of the minimum support.

In paper [1], the proposed algorithm shrinks original database by integrating user's domain knowledge, eliminating any non-related items and then mining k-frequent itemsets using the new database. It helps to decrease the number of times taken to scan the database and optimizes the process that generates candidate itemsets and facilitates support counting. This algorithm overcomes some of the weaknesses of the Apriori algorithm by reducing the number of candidate k-itemsets. Clustering and Graph based Association Rule [9] was proposed in which a cluster table is created by scanning the database and then the transactions are further clustered into clusters based on their length. Even though, the algorithm is scalable much time is wasted in constructing graphs for each cluster, which reduces the performance.

An Approximation based Incremental Memory Efficient Itemset Tree (AIMEIT) algorithm was introduced in this paper [2] which is an extension to Memory Efficient Itemset Tree (MEIT) algorithm to construct the itemset tree from data stream. The user defined minimum support of interest has been used along with Loss counting algorithm to prune transactions before inserting them into the tree. The proposed algorithm is more memory efficient and takes lesser processing time for constructing the tree. The outcome is the constructing of the tree and adds a pre-processing stage to prune the infrequent items from the data stream before inserting the transactions into the tree.

A mining algorithm called MAFIM algorithm [7] is proposed for frequent itemsets based on mapreduce and FP-tree. The advantage is that the MAFIM algorithm is fast and effective. Global frequent itemsets were got by mapreduce thus, promoting highly the efficiency of data mining is an effective outcome of this approach.

An optimization in the phase of generation-pruning of candidates by a new strategy for the calculation of frequent itemsets based on approximate values of supports that exacts the itemsets is introduced in this paper [5]. The new technique for mining frequent itemsets is based on the SupportMin. This value SupportMin in the pruning step of itemsets candidates determines if a k-itemset cannot be frequent. This technique decreases the execution time, when the support threshold increases for AprioriMin Algorithm.

In the Big Data era the need for a customizable algorithm to work with big data sets in a reasonable time becomes a necessity. This paper [6] proposes a new algorithm called Parallelizable Optimized Buddy Prima Algorithm (POBPA) for frequent itemset discovery that could work in distributed manner with big datasets. The approach is based on the original Buddy Prima algorithm and

the Greatest Common Divisor (GCD) calculation between itemsets which exist in the transaction database. The proposed algorithm introduces a new method to parallelize the frequent itemset mining without the need to generate candidate itemsets and also it avoids any communication overhead between the participated nodes. The proposed approach could be implemented using map-reduce technique or Spark. The advantage of this method is that it reduces the computation time and it can handle many datasets.

Hybrid Frequent Itemset Mining (HFIM) is introduced in this paper [11] which utilizes the vertical layout of dataset to solve the problem of scanning the dataset in each iteration. Vertical dataset carries information to find support of each itemsets. The proposed algorithm is implemented over Spark framework, which incorporates the concept of resilient distributed datasets and performs in-memory processing to optimize the execution time of operation. The advantage of this method is that it optimizes the execution time of operation and the non-frequent items are removed from the original horizontal input data, reducing the data size and only the frequent itemsets are generated. Since the size of vertical dataset is smaller than the entire dataset, it consumes less storage and processing time.

Customer Purchase Behavior (CPB) has been introduced [10] and used for finding frequent itemsets using minimum scans, time and memory and the rules are generated. The subset invention process is used for the generation of frequent itemsets which reduces the intermediate tables. This approach reduces main memory requirement since it considers only a small cluster at a time. The purchase behavior of the customer can be easily judged by using the Quine-McCluskey method. This algorithm is very efficient because of redundancy elimination and rule generation.

Two parallel incremental FIM algorithms called IncMiningPFP and IncBuildingPFP [14] are implemented on the MapReduce framework in this work. These algorithms preserves the FP-tree mining results of the original pass, and utilizes them for incremental calculations. It improves the efficiency of incremental FIM and saves large amount of tree mining time. It is efficient for solving incremental FIM problems. These algorithms achieve significant speedup over PFP (Parallel FP Growth).

An Incremental Frequent Itemset Mining algorithm [16] based on Apriori is proposed to deal with generating association rules based on available knowledge and the newly added data set only. Considering the time-consuming characteristic in processing large scale data set, Spark is implemented. It reuses existing results from previous computation to modify the frequent itemsets according to the newly added data, which avoids massive re-computation. The major advantage is that it avoids reduplicated computation and improves the performance of frequent itemset mining with no additional storage overhead.

The paper [17] uses an adaptive sliding window based strategy for mining the main frequent itemsets on streaming data. The key idea is to dynamically adjust the size of sliding window by exploiting the time-varying feature of streaming data. The proposed algorithm can enhance time performance by reducing the data size for mining. Using the sliding window algorithm, the main frequent itemsets are determined thereby achieving high efficiency, accuracy, and performance, which is the key metrics to be applied in computer forensics.

A parallel frequent itemsets mining algorithm called FiDoop [18] using the MapReduce programming model is proposed in the paper. In FiDoop, three MapReduce jobs are implemented to complete the mining task. The proposed solution is efficient and scalable. It achieves compressed storage and avoids the necessity to build conditional pattern bases. The outcome shows that it improves the performance of FiDoop by balancing I/O load across data nodes of a cluster. Designed and implemented FiDoop-HD an extension of FiDoop that efficiently handles high-dimensional data processing.

The weight judgment downward closure property for the weighted frequent itemsets and the existence property of weighted frequent subsets are introduced and proved first by the paper [20]. Based on these two properties, the Weight judgment downward closure property-based FIM (WD-FIM) algorithm is proposed to narrow the searching space of the weighted frequent itemsets and improve the time efficiency. Moreover, the completeness and time efficiency of WD-FIM algorithm are analyzed theoretically. Finally, the performance of the proposed WD-FIM algorithm is verified on both synthetic and real-life data sets.

For the purpose of efficiency improvement and resource saving, the paper [19] proposed an approximate variation of Eclat algorithm which is based on MinHash technique. The proposed algorithm called HashEclat considers the tradeoff between accuracy of the mining results and algorithm execution time. The theoretical analysis and experimental studies all indicate that HashEclat has the high performance in association rule mining and it can output almost all the frequent itemsets with faster speed and less memory space.

In the current works, however numerous algorithms were proposed to lessen space and proficiency a significant drawback was tended to. A large portion of the calculations involved more space or produce numerous candidate itemsets. In comparison with existing works, our algorithm BIT VECTOR MINING will reduce these issues and the Performance is impressively expanded. Henceforth, the checking cost is likewise decreased by utilizing our new proposed calculation.

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887
Volume 7 Issue III, Mar 2019- Available at www.ijraset.com

III. PROPOSED SYSTEM

Bit Vector Mining Algorithm scans the transactions database only once and develops a table as a two dimensional array where the columns portray the items in the transaction and the rows represent Transaction ID's. The table consists of bits (0 or 1) to indicate the presence or absence of an item.

1 indicates the presence of an item in a transaction and 0 represents the absence of an item in a transaction. The built table now consists of bit vectors for all individual items 'n' the transactions. The number of 1's indicates the presence of an item in a transaction. The number of 1's multiplied by total number of items gives the support threshold of each item.

If the support threshold value is greater than the minimum support threshold value, then the item is considered to be frequent 1-itemset. The table is updated with only frequent items. Frequent 2-itemset is determined by doing logical AND between each pair of consecutive frequent 1-itemset. Frequent 3-itemset is determined by doing logical AND between each pair of consecutive frequent 2-itemset. Now, the table is updated (i.e.) all the transactions with 2-itemsets are removed and the table will consist of transactions with k 3, where k is the total number of items. Thus after applying logical AND, the table is updated till N-1 transactions where N is the total number of transactions.

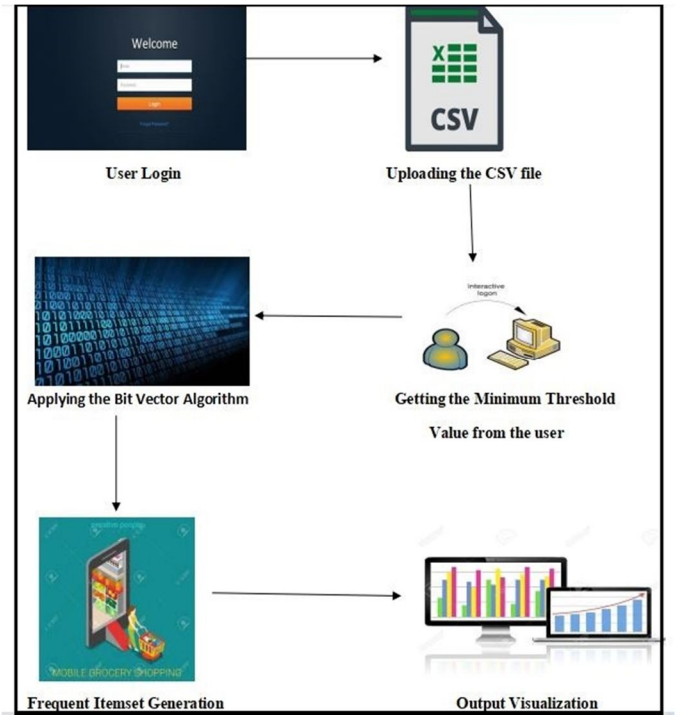

Fig. 3.1 – Architecture of the Proposed System

A. Bit Vector Mining Algorithm

1) Input: Temporal Database, TD

2) Output: Frequent Itemsets

a) Step 1: Create a table for the given transaction data set.

b) Step 2: Convert the itemsets in the table into (0s or 1s) bit vectors.

c) Step 3: Read the minimum threshold value from the user.

d) Step 4: Find the frequent item sets for 1 to n items (Perform logical AND operation to find frequent items for pairs of items until K - frequent itemsets).

e) Step 5: Calculate the support count for 1 to n items in a new table. Support count = (occurrence of 1's) ∗ (Total Number of items)

f) Step 6: If Support count > minimum threshold value then print the frequent item set. Else remove the item.

g) Step7: Create a table for the frequent item sets.

h) Step 8: Repeat from step 4 until there are no items left in the table.

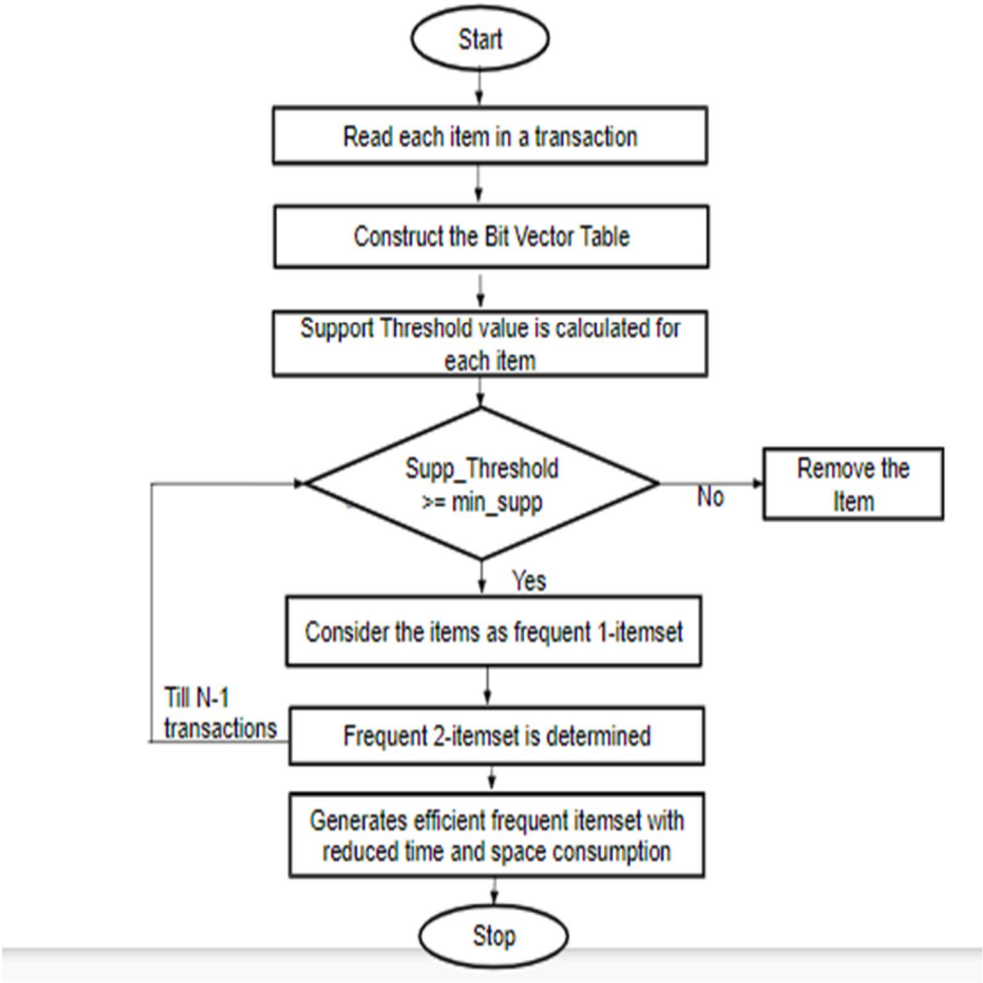

Fig., 3.2 - Frequent Item Set Generation Flowchar

IV. CONCLUSION

In some applications it is necessary to handle large volume of data. In such situations, this project will provide better performance in terms of time and space complexity when it is used with temporal database for mining frequent items. In this paper, the proposed technique for mining Frequent Item sets from transaction database depends on the bit vectors. In any case, the proposed methodology will have the capacity to spare some memory. Since this algorithm uses only single scan, the number of database scans are reduced and hence the computation time taken is also very less.

REFERENCES

[1] Altameem A & Ykhlef M (2018), 'Hybrid Approach for Improving Efficiency of Apriori Algorithm on Frequent Itemset', IJCSNS, 18(5), 151.
[2] Bai P & GK R. K (2016), 'Efficient Incremental Itemset Tree for approximate Frequent Itemset mining on Data Stream', IEEE 2nd International Conference on Applied and Theoretical Computing and Communication Technology, pp 239-242
[3] Bashir S, Halim Z & Baig A. R (2008), 'Mining fault tolerant frequent patterns using pattern growth approach', IEEE International Conference on Computer Systems and Applications, pp. 172-179.
[4] Djenouri Y, Djenouri D, Lin J. C. W & Belhadi A. (2018) 'Frequent Itemset Mining in Big Data With Effective Single Scan Algorithms', IEEE Access, 6, 68013-68026.
[5] Essalmi H, El Far M, El Mohajir M & Chahhou M (2016), ' A novel approach for mining frequent itemsets: AprioriMin', 4th IEEE International Colloquium on Information Science and Technology, pp 286-289.
[6] Gawwad M. A, Ahmed, M. F & Fayek M. B. (2017) 'Frequent itemset mining for big data using greatest common divisor technique', Data Science Journal, 16:25, pp 1-10.
[7] He B, Pei J & Zhang H (2017), 'The Mining Algorithm of Frequent Itemsets based on Mapreduce and FP-tree', IEEE, International Conference on Computer Network, Electronic and Automation, pp 108-111.
[8] Kamepalli S, Kurra RR & Krishna Y S (2016), 'A Multi-Class Based Algorithm for Finding Relevant Usage Patterns from Infrequent Patterns of Large Complex Data', Indian Journal of Science and Technology, 9(21).

[9] Krishnamurthy M., Kannan A., Baskaran R. & Kavitha M. (2011) 'Cluster based bit vector mining algorithm for finding frequent itemsets in temporal databases', Procedia Computer Science, 3, 513-523.
[10] Krishnamurthy M., Rajalakshmi E, Baskaran R & Kannan A (2013), 'Prediction of customer buying nature from frequent itemsets generation using Quine-McCluskey method', 397-411.
[11] Sethi K. K & Ramesh D (2017) 'HFIM: a Spark-based hybrid frequent itemset mining algorithm for big data processing', The Journal of Supercomputing, 73(8), 3652-3668.
[12] Shah A & Halim Z (2018) 'On Efficient Mining of Frequent Itemsets from Big Uncertain Databases', Journal of Grid Computing, 1-20.
[13] Sinthuja M, Puviarasan N & Aruna P (2015), 'Research of Improved FP-Growth (IFP) Algorithm in Association Rules Mining', International Journal of Engineering Science Invention, 2319-6734.
[14] Song Y. G, Cui H. M & Feng X. B (2017), 'Parallel Incremental Frequent Itemset Mining for Large Data', Journal of Computer Science and Technology, 32(2), 368-385.
[15] Yang Q, Fu Q, Wang C & Yang J. (2018) 'A Matrix-Based Apriori Algorithm Improvement', IEEE Third International Conference on Data Science in Cyberspace, pp 824-828.
[16] Yu M, Zuo C, Yuan Y & Yang Y (2017) 'An incremental algorithm for frequent itemset mining on spark', IEEE, 2nd International Conference on Big Data Analysis, pp 276-280.
[17] Xiong A, Huang Y, Wu Y, Zhang J & Long L (2018), 'An Adaptive Sliding Window Algorithm for Mining Frequent Itemsets in Computer Forensics', IEEE 17th International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering, pp 1660-1663.
[18] Xun Y, Zhang J & Qin X (2016), 'Fidoop: Parallel mining of frequent itemsets using mapreduce', IEEE transactions on Systems, Man, and Cybernetics: systems, 46(3), 313-325.
[19] Zhang C, Zhang X & Tian P (2017), 'An approximate approach to frequent itemset mining', IEEE Second International Conference on Data Science in Cyberspace, pp 68-73.
[20] Zhao X, Zhang X, Wang P, Chen S & Sun Z (2018), ' A Weighted Frequent Itemset Mining Algorithm for Intelligent Decision in Smart Systems', IEEE Access.