AN EFFECTIVE HASH-BASED ALGORITHM FOR FREQUENT ITEMSET MINING BY TIMESERVING PROJECTION

K. Lokeshwari[1], K. Shankari[2]

Abstract - *In these days, scrutinizing data is vital role in making the decision for producing information from database. Those information are mining from the database is very crucial and arduous phase. while acquiring information it should be most pertinent one to which is getting associations between various attributes. The acquired information that is hidden patterns will get more time to be mined when the range of data get increase and we need huge memory to run this process by the algorithm, which cause memory consumption due to heavy computation. To overcome above issue, an efficacious algorithm is needed which is hash based timeserving projection mining by single scan. This algorithm helps to fetch the information within one scan from the database by using hash key value for ordering those attributes to organize those itemsets into multiple hashes with the support of logical predecessor.*

Keywords - ***Frequent itemsets, Association rule mining, Lexicographic Order, Multi-Hash.***

I. INTRODUCTION

From the last decade, mining the frequent item pattern from the database has one of the crucial parts. Actually, this task was started along Market Basket Analysis (MBA) and the process of MBA to mining the data which are consider as transaction data, to make an analysed view of shopping behaviour of people to analyse the customers choices of supermarkets, e-shopping and telecom field. From the analysis (MBA), we can conclude the desired choices of items that are utilized by the people by frequently. In this step, the conceptualization of frequent item set mining occurred which highly forced to develop a huge count of efficacious algorithm to retrieve the data from the well-organized database. There are plenty of algorithms are available to fetch the frequent item set pattern, there are APRIORI, ECLAT and FP-growth which are utmost extensively acknowledged. The item sequences are easily expanding from item set, but we need tiny consideration in open up a new arrival application area which are genome mining and temporal pattern extraction from data base. In this concept, every time frequent item pattern to be mine from the database, it has the crucial part, which are the initial point of doing technical process. This process is mainly used to elucidate idea about what is data mining and also not a difficult to implement.

Now it's time to get an idea about what is an itemset? This word mainly used for data mining; it doesn't occur in regular dictionary. It means, the representation of set of items or an element/attributes which are grouped to represent as a single entity. In our research, we are talking about frequent item set which occurs many times in multiple transactions.

Further, the paper divided into upcoming phases, as phase II talks about the terminologies used in our approach, phase III describes the literature study of the related topics, and phase IV talks about the algorithm used to build our proposed research, in phase V the results and discussion describes our proposed, and finally phase VI commit the conclusion and future scope.

II. TERMINOLOGIES

1. Association Rule Mining

From every transaction in the database, we can identify the relationships between different items/attributes. It strongly identifies the rule which are discovered in the databases by using various scale of interestingness. It discovers the regularities within the products in large-scale transaction database by using point-of-sale (POS) which are widely used in supermarkets. The above-mentioned information is utilized for various purpose like target marketing, because the consumers buying patterns can be summarize from one another.

Let we assume S be the set of items in the database where the transaction occurs.

Consider the transaction $T \subseteq S$, contains set of items which are come together in one operation. The concept of association rule occurs between the group of items $Y \subseteq S$ and another set is $Z \subseteq S$, both are represented as $Y \Rightarrow Z$. It tells that repeated number of times the items occurred in the Y transaction also indicates a strongly possibility of the count of occurrences of set of similar items Z.

The soundness of this rule, is to indicate by the metric functions they are, *Support* and *Confidence* terminologies to implement the Market Basket Analysis. These terms are helps to identifying the joint participation of purchase and tells the occurrences of associations between products. Support represents the popularity of that product of all product transactions. Confidence can be concluded as the chances of purchasing both the products Y and Z.

Confidence is summed as the occurrences count of transactions that include both Y and Z fractioned by the occurrences count of transactions include only product Y. whereas k is refer to an itemset with k items. These terms are helping to identifying the joint purchase that is, customers always used to purchase together and beside with products associations between them.

Support always used to indicate that having popularity of that product in product transactions. The product support is summarized by the correlation of the sum up of transactions includes that product along with sum count of transactions.
Confidence can be explained as the possible way of purchasing both the products Y and Z. Confidence is calculated as the sum up of the transactions include both Y and Z divided by the sum up of transactions includes only product Y. We shall refer to an itemset with k items.

The Association rule mining has issues that is, we have to seek the rules to reach the above minimum level, that is threshold of support and confidence. The algorithm has two phase procedures, in the top most phase at very first we finding itemsets that are repeatedly occurred. Which itemset are frequent, it should satisfy the users defined minimum support requirement. In the upcoming phase, it uses the frequently occurred itemset which are produced in the previous phase, it led to generate all the specified rules to satisfy the minimum confidence.
In our concept, we are absorbing only the first phase, that is finding the itemset which are frequently occurred and computing their occurrences from the database.

2. Hashing

The concept of hashing, is mainly designed to rectify the complication of wanting to efficaciously find or store an item in a collection. That is, transformation of a continuous character that is, string into original string.

The main concept is, to seek the item fastly using hashed key than to finding the original value. This helps to index and retrieve items from the database. This function is to search to map the data of random size to keys of size, with little difference in data which are getting as input, it creates very huge differences in the produced output data. The group of output value produced by the function, called as hash values, codes. The function generates a table format data or database lookup by diagnosing duplicate copy of records in a file. It has a search key value to quickly map a data record in hash tables. This function process has to map the search key value to point an index, this index tells the place in hash table where the record stored actually.

It has two major features, they are

- Efficaciously generate the large itemsets.
- The transaction database size can be managed well.

In our proposed, we have to use of hash value as once to accumulate the calculated data in the table which has collection of hash values and when we are need of data it creates the way to be easy and fast to fetch from the table which has hash values. While saving those data we will save the data in a key value pair itemset and support count pair and these key pairs are sorted in a lexicographic order, later this order helps to access support count of any itemset at the time of retrieving.

III. RELATED WORK

Agrawal et. al [6] in this proposed, it has 3 algorithms namely aproiri, aproiriTid and aproiriHybrid which mainly used to seek the frequent item set. The aproiri algorithm initially started the process by the individual item set and it produce the large subset which is based on threshold value. In the succeeding passes, sown the largest subset which is decisive in the previous pass and determined the candidate item set. The aprioriTid does not count the support after the previous pass from database. To determine the support, an encoding of candidate item was used, to reducing the reading efforts. The aproiriHybrid, algorithm which produced by later combined the best features of the above two algorithms, which reduces the time for calculating candidate item sets.

Hai et. al [7] proposed a method talks about the compact of FP (Frequent Pattern) data structure. The smaller pattern base generate at the time of FP tree was scanned at only once a time. To get all occurred frequent items, the Frequent Pattern Mining have to performed recursively. The required number of database scan occurred to build the FP tree.
Agrawal et. al [11] proposed an algorithm set to focus on long pattern in database. This algorithm utilized the lexicographic order tree and depth first strategy to process. The lexicographic order tree used to represent the item set in an organized order. Mainly the depth first search used to make a number of count the support of a candidate node and along reducing time for support count, especially the main use of this is, to seek long patterns very faster.

Agrawal et. al [12] proposed an algorithm purely depends on lexicographic tree order. This tree structure helps to reduce the time taken for CPU for counting the frequent item set. The matrix count technique used to mine the frequent items and the transaction at each node of tree was in the hierarchical structure of tree are successively projected.

Lin et. al [13] proposed a algorithm in a level wise for mining the associations between candidate sets and used to seek the item sets which are frequently occurred in one pass by pass. The kernel process gets push up, to stores the itemset as lexicographic order into a hierarchal tree, it called as Lextree.

Rao et. al [14] proposed a double hashed algorithm for the itemset mining which are occurred frequently, where data are stored in the form of vertical format from transaction database. The collision issues will be avoided by using double hashing technique.

In our suggested approach, we are using multiple hashing to arrange all the item set in a well arrange logical structure, which elicit the relevant information in a single database scan to be practiced in the latter steps of association rule mining.

IV. PROPOSED APPROACH

A. Algorithm Constructs

Most of the studies were done regarding frequent itemset mining. Various algorithms have developed thus far, but till now little issue occurred, that is, enormous number of itemset generation that are frequently occurred and maintaining those records is a complicated task and also calculating the occurrences count of any attribute occurrence for computing support in the database, confidence or any other interestingness parameters.
In our proposed algorithm focuses on not only seek the frequent itemset but also evaluate the count of occurrences of any number of attribute or any attributes combination in the database that too in a single scan.
In our proposed approach, the attributes are arranged in the various hash table with key pair value. For each attribute, the separate hash table have maintained and the combinations with their counts, which tells the occurrences count in the database. These hash tables are combined together to form a hierarchal manner. It has different levels, each level has gradual increasing of attributes, i.e., level 1 has single attribute of itemset, level 2 has comprised all the combinations of 2 attributes and so on in N levels, where N identifies the number of properties.

B. Multi-hash Structures

For each level there will be available hash tables, those are connected with their respective attribute at level 1.
a) At level 1: For every single itemset, one hash value available. It consists of each attribute and the number of occurrences in the database. Those hash entries are connected with their respective hashes at each level.
b) At level 2: in level 2, if attributes are (n) numbers, then (n-1) hash tables available in this level. Each hash table has different attributes and it has combinations of 2 attributes along with number of occurrences count in the database.
For example: Let there are attributes A, B, C, D in the database.
Level 2 will contain 3 hash tables corresponding to A, B, C attributes. Hash table (hashA2) will contain combinations AB, AC, AD.
c) At level 3: In this level, (n-2) hash tables available. Each hash table has different attributes and it has combinations of 3 attributes along with number of occurrences count in the database.
For example: Level 3 will contain 2 hash tables corresponding to A, B attributes. Hash table (hashA3) will contain combinations ABC, ABD, ACD.
d) At level n: this level has only one hash table with the combinations all n attributes.
For example: Level 4 will contain 1 hash table (hashA4) with one combination corresponding to ABCD itemset. Structure for a four-attribute database is shown in Fig 2. Where HASH1 indicates the hash table at level1.
HASH2A, HASH2B, HASH2C is the hash tables at level 2 of respective A, B, C attributes. HASH3A, HASH3B are the hash tables at level 2 of respective A, B attributes.

C. Counting Combinations

The counter value will be increased when the transaction is scanned of each possible combination of that itemset.
For example: if there is transaction corresponding to one customer purchased like ACD together, then the transaction in level 1 hash A, C, D will be incremented by 1.
Then at level 2 AC, AD will get incremented in the first hash table. And CD will get increase in the third hash. And level 3 ACD will be incremented.

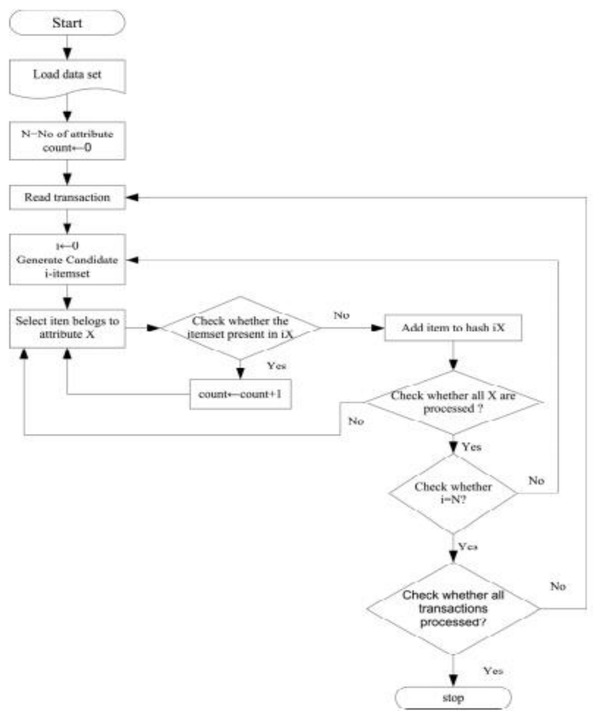

FIG 1: *Algorithm Flowchart*

The process flow of this algorithm is served as in the above FIG 1. Initial step is to begin, based on attribute in the database, the hash structure will build. Once created empty hash structure, in the next step will begin where the scanning process will occur as row by row. In database every action considers as transaction, which are indicated as row. For every transaction, itemset count will be incremented into their corresponding locations. The above process will continue till the scanning process completely done at once in the database.

Pseudo code of this proposed algorithm (LFIG) is given below.

Algorithm: LFIG (Lexicographic Frequent Itemset Generator**) Input:** Dataset D;
Output: Hashing of frequent itemsets;

```
1. Begin
2. for i=1 to N // no. of tuples in dataset D;
3. for each tuple Ti // j є {positive integer}
4. for k=1 to n // no. of attribute Ti;
5. for each attribute x
6. // x є {set of attributes of Ti}
7. If (k-itemset = = HASH_k_x ())
8. HASH_k_x. count + +;
9. Else
10. HASH_k_x HASH(k-itemset);
11. HASH_k_x. count + +;
12. End for
13. End for
14. End for
15. End
```

Table - 1: *Computation Cost of various Functions*

OPERATIONS	COMPUTATIONAL TASK
Insertion (Regarding one transaction)	O(2k-1)
Insertion (Whole Database Scan)	O(N*2k-1)
Updation (of one itemset)	O (1)
Searching (for any itemset)	O (1)

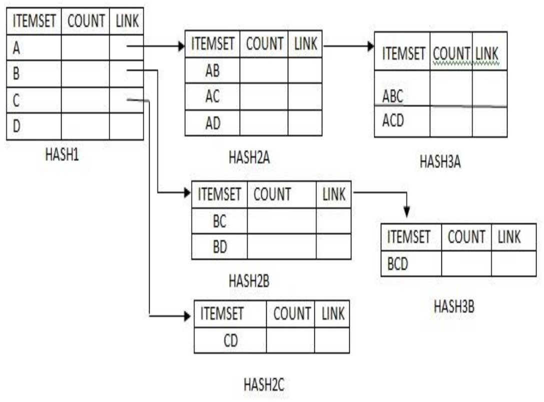

FIG 2: *Multi-Hash Structure*

In Fig 3, the below table represents the transaction database as a sample data, where TID represents the transaction id and another column is transaction, which tells the occurrences of the attributes in the transaction. This sample is examined to acquire four attributes like
A, B, C, D in lexicographic order and it has six sample transactions.

Table - 2: *Tractional Database*

TID	TRANSACTION
1	ABC
2	ABD
3	AB
4	BC
5	BCD
6	ABCD

The first transaction of this sample database, after scanning it has view of attribute A, B, C appeared together. The occurrences of possible itemsets are as follows, which are categorized as level wise. They are, {A}, {B}, {C} at level 1, {AB}, {AC}, {BC}at level 2 and {ABC} at level 3 all these itemsets are incremented at their respective levels and the status of the structure after scan of 1 transaction is shown in Fig4.

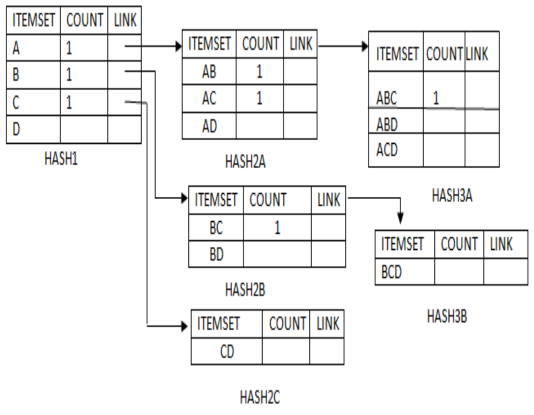

FIG 3: *Hash after the first transaction scan*

The final status of structure of the hash will be somewhat like shown in Fig5 once scanning all transactions of sample database.

This has all the itemsets well arranged with their respective counts. After final construction it is accessible for anyone to get any information regarding occurrences of any itemset in O (1) computational cost.

V. RESULTS AND DISCUSSIONS

The ultimate aim of mentioned algorithm is to achieve the maximum information from itemset database in a lone database scan, it will be utilized when processing with the association rule mining. The above process gives the clear view of information in one scanning from database which tells that we are efficacious to deliver the all possibility of itemset with their occurrences along with their respective counts in the database. in the time of scanning the one transaction from database, if it has K attributes, then it will have 2^k -1 possible itemsets, it has group of combinations of all kinds of attributes of the datasets, and itemset have to be updated at the time of entering in to the database.

The cost of transaction for one scanning will be $O(2^k-1)$. The total cost of N transaction for total scanning is $O(N*2^k-1)$. Once scanning the all transaction from the database it will cost only O (1) to searching anything from database, it has hash structure which is the property of hash data structure that searches from hash with in O(1) time.

Here the K represents the number of attributes in a single transaction.

VI. CONCLUSION AND FUTURE SCOPE

This paper talks about the peculiar accession to seek the maximum information from a database respecting to frequent itemset and their counts with in a single database scan. We have been used vitality of the hash tables with the respective of lexicographic organize of attributes. In our proposed we have a hash structure and it has multiple hash tables which are arranged in a multiple level, each level contains the itemset of k-th itemset along with hash values. This technique serves to organize lots of itemset into a well- organized structure which make easier and fast to access later for getting any kind of combinations of attributes.

The one crucial drawback of this concept is, memory space that is, it needs more space to store all possible itemsets count initially at the time of generation. In future, next proceedings will be continued by considering this space issues.

REFERENCES

[1] P. T. S., P. Mayur, L. Dhara, K. Jahnvi, D. Piyusha, P. Ashish and P. Reecha, "An Analytical Study of Various Frequent Itemset Mining Algorithms," Computer and Information Technology Sciences, vol. I, pp. 6-9, 2013.

[2] R. R. Naik and P. J.R. Mankar, "Mining Frequent Itemsets from Uncertain Databases using probabilistic support," International Journal of Emerging Trends & Technology in Computer Science, vol. II, no. 2, pp. 432-436, 2013.

[3] R. Ahirwal, N. K. Kori and D. Jain, "Improved Data mining approach to find Frequent Itemset Using Support count table," International Journal of Emerging Trends & Technology in Computer Science, vol. I, no. 2, pp. 195-201, 2012.

[4] R. C. Agarwal, C. C. Aggarwal and V. Prasad, "A Tree Projection Algorithm for Generation of Frequent Itemsets," Journal of Parallel and Distributed Computing, vol. 61, no. 3, p. 350 – 371, 2001.

[5] J. s. park, M.-s. chen and P. s. yu, "An effective hash-based algorithm for mining association rules," in 1995 ACM SIGMOD international conference on Management of data, San Jose, CA, USA, 1995.
[6] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," in 20th International Conference on Very Large Data Base, Santiago, Chile, 1994.

[7] J. Han, J. Pei and Y. Yin, "Mining Frequent Patterns without Candidate Generation," in 2000 ACM SIGMOD

international conference on Management of data, Dallas, TX, USA, 2000.

[8] M. J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li, "New Algorithms for Fast Discovery of Association Rules," in Knowledge discovery of databases-97, vol. 97, pp. 283-286, 1997.

[9] J. Wang, J. Han, Y. Lu and P. Tzvetkov, "TFP: An Efficient Algorithm for Mining Top-K Frequent Closed Itemsets," IEEE Transactions On Knowledge And Data Engineering, vol. 17, no. 5, pp. 652-664, 2005. s

[10] J. Liu, Y. Pan, K. Wang and J. Han, "Mining Frequent Item Sets by Opportunistic Projection," in eighth ACM SIGKDD international conference on Knowledge discovery and data mining, Edmonton, AB, Canada, 2002.

[11] R. C. Agarwal, C. C. Aggarwal and V. V. V. Prasad, "Depth First Generation of Long Patterns," in sixth ACM SIGKDD international conference on Knowledge discovery and data mining, Boston, MA, USA, 2000.

[12] R. C. Agarwal, C. C. Agarwal and V. Prasad, "A Tree Projection Algorithm for Generation of Frequent Itemsets," Journal on Parallel and Distributed Computing (Special Issue on High Performance Data Mining), vol. 61, no. 3, pp. 350 - 371, 2001.
[13] M. Yen and S. Y. lee, A Fast Leicographic Algorithm for Association Rule mining in Web Applications, ICDCS Workshop of Knowledge Discovery and Data Mining in the World-Wide Web, 2000.

[14] N. G. Rao and S. Aguru, "A Hash based Mining Algorithm for Maximal Frequent Item Sets using Double Hashing," Advances in Computational Research, vol. 1, pp. 1-6, 2012.