Association Rule Generation using Modified Hashing Function

M.Ramakrishnan
Professor & Head,
Department of Information Technology, Velammal Engineering College, Chennai-66, Tamil Nadu, India

D.Tennyson Jayaraj
Research Scholar, Department of Computer Science, Manonmaniam Sundaranar University, Tirunelveli, Tamil Nadu, India

ABSTRACT
Association rule mining is one of the most interesting and challenging task in data mining process. There exists many association rule mining techniques, each having merits and demerits. The main problem that exists in many traditional association rule mining algorithms is that these algorithms need more than one database scan to generate association rules. As scanning the database is a costly operation, algorithms capable of generating association rules with only one scan is the need of the hour. In this paper, a novel algorithm for generating association rules is presented which uses hashing function. This algorithm scans the database only once by utilizing the latest version of priori algorithm, direct hashing algorithm and pruning process. The algorithm discovers set of association rules from frequent k-item sets by computing the frequency of each item set. Then pruning process is applied to minimize the number of item sets generated after scanning the size of the database. Experimental results show that our method is very effective in generating association rules without any collision, leading to very high data accuracy.

Keywords
Association Rule Mining, ARM, Hashing, Pruning, Basket Market Analysis, Apriori Algorithm

1. INTRODUCTION
Association rule mining is the process of finding the rules that will predict the occurrence of an item based on the occurrences of other items in the transaction. The basic idea is to find the co-occurrence relationship among data items called associations[1]. Many real world applications are using association rule mining to find the interesting and useful rules from a large transaction data. One important application of association rule mining is the application in Market Basket analysis which aims to discover how items are purchased by customers in a super market and their associations[2]. Association rule mining algorithm can be broadly classified as

(i) Apriori Algorithms

(ii) Partition Algorithms

(iii) Sampling Algorithms

Apriori algorithms are based on main memory, which imposes a limitation on the size of the dataset to be mined. The method works in two steps by generating frequent item sets in the first step and generating association rules in the second step. The first step consists of two phases with candidate item set generation in the first step and frequent item set generation in the second step. The method scans the database multiple times and it is not possible to find the number of database scans needed earlier[3].

Partition algorithms, on the other hand, are based on dividing the database into number of non-overlapping partitions. Frequent item sets local to the partition are generated for each partition. Partition algorithm need minimum of two database scans with generation of frequent item sets in the first scan and generating global item sets in the second scan[4]. In partition algorithms, a special data structure called TIDLIST is used which contains transaction IDs of all the transactions corresponding to an item set in the partition[4].

Sampling algorithm picks random samples from the database and tries to find frequent item sets in the samples. Finding frequent item sets is based on using support that is less than the user specified minimum support for the database. Then the algorithm also finds candidate item sets that did not satisfy minimum support. Performance of this algorithm relies on the quality of the sample chosen. [5]

Chin-Chen Chang et al [6] proposed an efficient algorithm for incremental mining of association rules. Incremental algorithms can manipulate earlier mining to get final mining outputs. The algorithm uses backward approach and scanning incremental database. Instead of scanning original database for frequent item sets, occurrence counts of newly generated frequent item sets are accumulated and infrequent item sets are deleted. The running time of NFUP is directly proportional to transaction number of incremental database.

Iko Pramudiono and Masaru Kitsuregawa [7] proposed tree structure based generalized association rule mining algorithm called FP-tax and it employs a tree structure to compress the database. Two methods are used to traverse the tree structure viz. bottom up and top down. This algorithm overcomes one of the most complicated mining tasks that require long processing time.

Ya Han Hu and Yen Liang Chen [8] proposed an algorithm for mining association rules with multiple minimum supports. The algorithm is the improvement of traditional apriori based MSapriori (Minimum Support) algorithm proposed by Liu et a l[8]. The proposed algorithm is two fold : with MIS-tree construction to store the crucial information about frequent patterns in the first step. In the second step, appropriate thresholds for all items at a time are set. Generally, users tune item supports and run the mining algorithm repeatedly till a satisfactory value is reached.

Farah Hanna Al-Zawaidah et al [9] proposed an improved algorithm for mining association rules in large databases. Key challenge in developing association rule mining algorithm is

that rules generated in extremely large databases makes algorithm inefficient. Further, understanding the generated rules by the end users is difficult. The algorithm presented is derived from conventional apriori approach with additional features.

A.Zemirline et al [10] proposed an efficient association rule mining algorithm for classification. The algorithm name is Association Rule Mining algorithm for classification (ARMC) and it extracts the set of rules, specific to each class. The algorithm uses fuzzy approach to select the items and it does not require the user to provide thresholds. This algorithm contain different features like covering all training instances and leaving no unclassified instances, requires only one pass to discover rules and uses novel model for building classification model. The features of this algorithm are not available in traditional associative classification methods.

R.Rathinasabapathy and R.Bhaskaran [11] presented the complexities in finding efficient association mining algorithm. Several variations of association rule mining algorithms exists such as Apriori algorithm that uses has functions in finding large 2-itemsets and 3-itemsets, direct search method for finding other large k-item sets. Éclat algorithm that uses perfect hash function in finding 2-itemsets and 3-itemsets, vertical method for finding other large k-item sets.

The main problem of many traditional association rule mining algorithms is that these algorithms need more than one database scan to generate association rules. As scanning the database is a costly operation, algorithms capable of generating association rules with only one scan is the need of the hour. This paper is organized as follows: Section 1 provides introduction about the association rule mining and section 2 provides preliminary work.

2. PRELIMINARY WORK

The problem of association rule mining can be stated as follows : suppose a file consists of a set of transactions T such that each transaction includes all the items I = { I1, I2, I3….. In}. A transaction t is said to contain set of items A, if and only if $A \in t$. An association rule is an implication of the form $A \Rightarrow B$, where A and B are set of items, $A \cap B = \varnothing$ and $A \subset I$, $B \subset I$. A or B is a set of items called item set. The support for the rule $A \Rightarrow B$ is the percentage of transactions in T that contains $A \cup B$ and can be estimated as $P(A \cup B)$, where P stands for probability. Support defines how frequent the rule is applicable in the transaction set T. If T contains n transactions, then the support for the rule $A \Rightarrow B$ is computed by

$$\text{Support}(A \Rightarrow B) = (A \cup B).\text{count} / n$$

Support is used to test whether a rule may occur due to chance or not. Confidence for the rule $A \Rightarrow B$ is the percentage of transactions in T that contains both A and B. It can be expressed as an estimate of conditional probability, P(A | B) where P represents conditional probability. Confidence for the rule $A \Rightarrow B$ is computed by

$$\text{Confidence}(A \Rightarrow B) = (A \cup B \text{ count} / A.\text{count})$$

This defines the predictability of the rule. Confidence is used to check whether one can reliably infer or not. The association rule mining algorithm generates all association rules that have support and confidence greater than or equal to user specified minimum support and minimum confidence. Association rule mining process contains two steps[12]. In the first step, frequent item sets that contain support and confidence above the user specified threshold are generated. Generation of association rules from frequent item sets is done in the second step. Let f is a frequent item set and we will find all the non empty sub sets of f. For every sub set x, the process generates rules of the form $x \Rightarrow (f - x)$. The generated rule is accepted if and only if the ratio of support (f – x) to support (x) is equal to or greater than the user specified minimum confidence level else the rule is rejected.

2.1 Direct Hashing and Pruning

Direct Hashing and Pruning, which is abbreviated as DHP is used to generate large item sets efficiently. DHP is having advantages such as effective reduction on transaction database size and effective reduction in database scan apart from its efficiency in generating large item sets. DHP uses hash function. DHP gathers information about candidate set in advance using hash function. Each bucket in the hash table contains an integer which represents number of item sets that have been already hashed to this bucket so far. Based on the output, a bit vector is constructed and value of the bit is set to one if the number of corresponding entry in the hash table is greater than or equal to minimum support.

The hash function is a black box which points to an address if a key is provided. H(k) is the hashing function which transforms key k into the address where the appropriate item sets are stored[13].

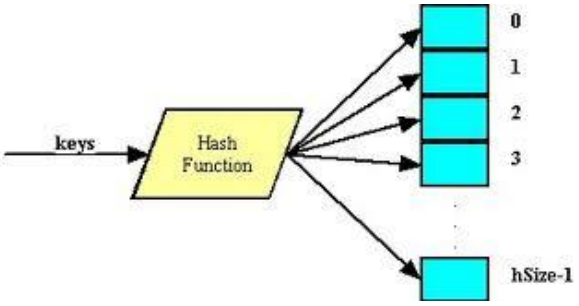

Fig 1. Hash Function

Using hash function directly for association rule mining generates certain draw backs also. Collisions may occur while using DHP as hashing function may address to two sub sets of each transaction for a single key k. that is in data mining terminology, counts in hash table can only be used to filter out the item sets whose numbers of occurrences are less than minimum support. If counts are greater than or equal to the minimum support, corresponding 2-item sets are candidate 2-item sets. The size of the candidate set may be close to that of Apriori if the rate of collision of hashing function is too high.

Apart from collision, DHP algorithm utilizes the fact that any sub set of a frequent item set must be frequent item set by itself.

3. PROPOSED METHOD

A new association rule mining algorithm is presented which finds the hash index for k-item sets. This algorithm overcomes the drawbacks listed in the above section. The algorithm uses both hashing and pruning algorithms. The basic idea of the proposed method is to divide the hash table into Hk in virtually iterative way. To overcome collision, we need to design a hashing algorithm that can spread the records over the available address. The efficiency of a hash function is

the measure of how efficiently the function produces values for elements within a set of data and complexity is represented by O(1). Hash function has to satisfy the following constraints

The data set is presented as continuous numeric data set.

(i) All the items in the item set are in sorted order (i.e. descending order such that $I_k<I_{k-1}<I_{k-2}$ ……. $I_2 < I_1$)
(ii) The number of distinct k – item sets must be equal to $\begin{bmatrix} n-r \\ k-1 \end{bmatrix}$ where n is the number of elements in the data set, 'r' is the smallest element in the data set.

To generate distinct item sets, we divide the hash table in to (n-k+1) parts were n represents number of items in each item set and k represents number of item sets. This is done in the first iteration. During the next iteration, previous iteration part is taken as parent part and divides it starting with first item of the parent part. This divides the has table to (n-[k-]+1). This procedure of division is repeated until we get 1-item set[14].

The start index for each part and the size taken are computed by using the equation

$$\sum_{i=1}^{X_{k-1}} S_i \text{ where } S_i = \begin{bmatrix} n-i \\ k-1 \end{bmatrix}$$

The next index is calculated using the equation

$$\sum_{i=1}^{X_{k-1}} S_i + \sum_{i=X_{k+1}}^{X_{k-1-1}} S_i$$

As optimal ratio is gained in our algorithm, collision will not occur. This is used to count frequency of k-item sets, which gives reliable result and actual index of each k-item sets.

3.1 Algorithm Enhancement

A specific data structure is used to implement our proposed algorithm. The data structure consists of root, after pointer, before pointer, valid bit array and nodes. Valid bit array represents numeric variable describing the frequency of each k-item sets[14]. Valid bit array size is N × K where N is the number of items. The first k-item set will be one which represents the k-item sets frequency in the hash table. Initially, root and tail pointers point to nil. First k-item set is computed. Now new node is created, root and tail pointing to the new node. Set the corresponding valid bit to 1.

Next, k-item sets are generated by computing index of k-item sets using our proposed method and check whether it contains valid bit array. If it is 1, it is in list search and increase the count by one else dynamic array list is inserted and counter is incremented by one.

New node is added to the end of the list if hash value greater than maximum value and tail pointer pointing to it. Corresponding valid bit is set to 1. The data structure contains next pointer which points to next adjacent node in the list.

3.2 Hash Class

Instance of a hash table is referred as hash class and it defines three pointers, with first pointer pointing the first node of the hash table, two other pointers are before and after which points to the previous and next node to a specific item set. When adding new k-item sets into the hash table, appropriate position is to be located so that the value can be retrieved using hash key. Increment index method computes the right position for an item set. The following are the main methods of hash class[11].

Searchnode method computes the position to insert the new node in the hash list by performing summation of valid bit array from beginning to the index parameter. Node is inserted by using Insert_node method and it works based on the particular scenario. If the hash table is empty, new node is created and root and tail pointing to it. Increment the counter by one and change valid bit to 1 for this item set[15].

If item sets are already available in the hashing table, then we can insert the new node as first node or add new node to the list. If it is inserted as first node then, get the position of already available first node, assign it to new node and increment the data by one. For the later case, just increase the data by one and change corresponding valid bit to one.

Valid bit array value is zero initially and when new k-item sets are added, the corresponding valid bit will be changed to one. This set up reduces the size of the hash table, minimizes the problem of unused space and avoids collision. This hashing technique can be used with all apriori algorithms like DHP.

4. ASSOCIATION RULE GENERATION

The algorithm works in two phases with scanning the database to compute frequency of each k-item set for all k and saving the item sets in the hash table in the first step. Generating interesting rules is done in the second step[16].

Phase I : Find all the k-item sets for each transaction of the database, compute index for each item set and increase corresponding index in hash table by one. Store them in array list. Once complete scan of database is over, perform the following :

1) delete all non frequent 1-item set from array list
2) delete all non frequent nodes for each item set in the hash table and change corresponding valid bit to 0.
3) prune k-item set by determining (k-1) item sets

Phase II : Find all the association rules by using frequent 1-item set

5. CONCLUSION

In this paper, generating association rules using perfect hash function is presented which is collision free and generates association rules with only one scan. More over, the proposed method does not require any complex operations. A new data structure is used to implement hash table which makes this algorithm cost effective. The data structure also reduces the size of normal implementation of any hashing technique. This enhanced algorithm can be very well used with other algorithms like PHP and DHP. Experimental results show that the generated rules are high in accuracy and the method does not suffer from collision problem.

6. REFERENCES

[1] Y. Yin et al., "Association Rules Mining in Inventory Database", Data Mining. © Springer 2011

[2] Lei Chen "The Research of Data Mining Algorithm Based on Association Rules", The 2nd International Conference on Computer Application and System Modeling (2012)

[3] Han, J., Kamber, M. & Pei, J. (2006). Data Mining: Concepts and Techniques, 2nd edition, Morgan Kaufmann.

[4] Hui Cao; Gangquan Si; Yanbin Zhang; Lixin Jia, "A density-based quantitative attribute partition algorithm for association rule mining on industrial database," American Control Conference, 2008 , vol., no., pp.75,80, 11-13 June 2008

[5] Kanakubo, M.; Hagiwara, M., "Speed-up Technique for Association Rule Mining Based on an Artificial Life Algorithm," Granular Computing, 2007. GRC 2007. IEEE International Conference on , vol., no., pp.318,318, 2-4 Nov. 2007

[6] R.Amornchewin and W.Kreesuradej "Incremental association rule mining using promising frequent itemset algorithm", In proceeding of Information, Communications & Signal Processing, 2007 6th International Conference on Data Mining

[7] Jong Soo Park , Ming-Syan Chen , Philip S. Yu, Efficient parallel data mining for association rules, Proceedings of the fourth international conference on Information and knowledge management, p.31-36, November 29-December 02, 1995, Baltimore, Maryland, United States [doi : 10.1145/221270.221320]

[8] Ya-Han Hu, Yen-Liang Chen, Mining association rules with multiple minimum supports: a new mining algorithm and a support tuning mechanism, Decision Support Systems, Volume 42, Issue 1, October 2006, Pages 1-24, ISSN 0167-9236,

[9] Farah Hanna AL-Zawaidah, Yosef Hasan Jbara, Marwan AL-Abed Abu-Zanona, " An Improved Algorithm for Mining Association Rules in Large Databases", World of Computer Science and Information Technology Journal, Vol. 1, No. 7, pp. 311-316, 2011.

[10] A.Zemirline, Lecornu, B.Solaiman, and A. Echcherif, "An Efficient Association Rule Mining Algorithm for Classification ", L. Rutkowski et al. (Eds.): ICAISC 2008, LNAI 5097, pp. 717 728, 2008. Springer, Verlag Berlin Heidelberg 2008

[11] Rathinasabapathy, R.; Bhaskaran, R., "Performance Comparison of Hashing Algorithm with Apriori," Advances in Computing, Control, & Telecommunication Technologies, 2009. ACT '09. International Conference on , vol., no., pp.729,733, 28-29 Dec. 2009

[12] Park, J. S., Chen, M. & Yu, P. (1995). "An Effective Hash-Based Algorithm for Mining Association Rules," ACM SIGMOD Record archive, 24(2), 175 – 186.

[13] Seiden, S. S. & Hirschberg, D. S. (1994). "Finding Succinct Ordered Minimal Perfect Hash Function," Elsevier Information Processing Letters, 51(6), 283-288.

[14] Sun, X., Li, M., Wang, H. & Plank, A. (2008). "An Efficient Hash-Based Algorithm for Minimal K-Anonymity," Proceedings of the thirty-first Australasian conference on Computer science, 01 01 January, Wollongong, Australia, 101-107.

[15] Tseng, M., Lin, W. & Jeng, R. (2008)."Incremental Maintenance of Generalized Association Rules under Taxonomy Evolution," Journal of Information Science, 34(2),174-195.

[16] Han, J., Kamber, M. & Pei, J. (2006). Data Mining: Concepts and Techniques, 2nd edition, Morgan Kaufmann.