Research article Available online www.ijsrr.org ISSN: 2279–0543

International Journal of Scientific Research and Reviews

Improved Hashing Technique in Assoiciation Rule Mining

L. Padmavathy

Department of Information Technology, Kg College of Arts and Science,Coimbatore,
Tamil Nadu, India
Email: padmavathy.l@kgcas.com

ABSTRACT

Association rule mining is one of the techniques to find out frequent itemsets. In this paper, a new hashing technique is introduced called H-Bit Array Hashing Algorithm (H-BAH) to find out the frequent itemsets. H-BAH technique uses the vertical format for finding candidate itemsets, which is placed in a linked list for finding frequent itemsets. It also avoids primary and secondary clustering and place the items in a collision free hash table.

KEYWORDS: Frequent item set, Hash table, Collisions

***Corresponding author**

L. Padmavathy

Department of Information Technology,

Kg College of Arts and Science,Coimbatore.

Email: padmavathy.l@kgcas.com

INTRODUCTION

Data mining is the process of sorting through large amount of information and picking out the relevant information. Data mining refers to using a variety of techniques to identify nuggets of information or decision-making knowledge in the database and extracting these in such a way that can be put to use in areas such as decision support, prediction forecasting, and estimation. It is also known as knowledge discovery process, knowledge mining from data, knowledge extraction or data/pattern analysis.

ASSOCIATION RULE MINING

Data mining[1] has many algorithms namely clustering, classification, association rule mining and more. Among these algorithms, Association Rule Mining (ARM) is one of the most important techniques and it is a method for discovering interesting relations between variables in large databases. It is initially used for Market Basket Analysis to find how the items are taken by customers frequently. A set of items is usually referred as "itemset", and an itemset with 'k' number of items is called "k-itemset". An association rule has two parts, an antecedent (if) and a consequent (then). An antecedent is an item found in the data and consequent is an item that is found in combination with the antecedent. The two important concepts of ARM are **support** and **confidence,** which is applicable in all transactional data

Support

The support of an itemset X is defined as the proportion of transactions in the dataset which contains the item set.

$$SUPPORT = \frac{SUPPORT\ COUNT}{NO.OF.TRANSACTION}$$

Confidence

Confidence is used to find whether the generated rules are strong or not. The confidence of a rule is defined as

$$CONFIDENCE\ (A \rightarrow B) = \frac{CONFIDENCE\ OF\ (A \rightarrow B)}{SUPPORT\ OF\ A}$$

Association Rule Mining is a two – step process

***Step 1*:** Find all sets of items (itemsets) whose support is greater than the user specified minimum support. Such itemsets are called frequent itemsets.

Step 2: Generate strong association rules from the frequent itemset. These rules must satisfy the user specified minimum support and minimum confidence.

The general form of an association rule is X => Y, where X and Y are two disjoint itemsets. For example, an association rule can be defined as

{ Potatoes } => { Burger }

The above rule has support 4% and confidence 50%. It indicates that 4% of all transactions contain both the items, and 50% of customers who purchased potatoes, is also liked to buy hamburger meat. The support count refers to the number of transactions has that particular item.

RELATED WORKS

Association Rule Mining (ARM) has been improved by various hashing techniques **(**Chin-Chen Chang and Chih-Yang Lin 2005, Gangadhara Rao and Srisha Aguru 2012, John and Soon, 2002). This section describes about different Hash-Based techniques which are used to increase the efficiency of ARM.

[Ayse Ozel and Altay Guvenir, 2001] proposed Perfect Hashing and Pruning [PHP] algorithm to generate the frequent itemset of a transaction database. The PHP algorithm has some of the features of Direct Hashing and Pruning [DHP] algorithm. The PHP[13] employs hashing facility, to keep the actual count of occurrence of each candidate itemset of the database. It also prunes the transactions which do not contain any frequent items, and trims non-frequent items from the transactions at each step in order to improve the accuracy of rules.

[Chin-Chen Chang and Chih-Yang Lin, 2005] presented perfect hashing schemes for mining association rules. The Direct Hashing and Pruning [DHP] algorithm that utilizes hash table in identifying the validity of candidate itemsets according to the number of the table's bucket accesses. The hash table used in DHP [4] is plagued by the collision problem.

[Don-Lin Yang et al., 2001] proposed an efficient technique called Hash-Based Method for Maximal Frequent Set [HMFS], for discovering maximal frequent set. The HMFS method combines the advantages of Direct Hashing and Pruning [DHP] algorithm and Pincher-Search algorithm. Hash technique of DHP algorithm is used to filter infrequent itemset in bottom-up direction. Then a top-down technique which is similar to Pincher-search algorithm is used to find the maximal frequent itemset. By combining the advantage of DHP and Pincher-Search algorithms, number of database scan and the search space of items are reduced.

[Ebrahim Ansari Chelche et al., 2010] presented hashing technique in the Fast Distributed Mining [FDM] algorithm. Distributed environment consists of several dataset. For each local dataset items are placed in hash table by using a hash function. Local frequent items are mined by using local minimum support. Using Distributed Hash Filtering, count of items in all local buckets are added and stored in a global hash table. Global frequent items[5] are mined by using global minimum support, which is greater than local minimum support value.

[Gangadhara Rao and Srisha Aguru, 2012] proposed double hashing method to find out the frequent itemset in hash table. Double hashing technique stores the itemset in buckets based on hash function. At the time of hash collision, double hashing technique is used to resolve them. After hashing candidate 1-itemset, frequent 1-itemset is calculated by using minimum support[6], and then candidate 2-itemset is hashed and so on. Maximal frequent items are directly calculated from frequent items itself. Finally from all the frequent items, association rules are generated. The drawback of this technique is using two hash functions and its memory usage is also high.

[Hassan Najadat et al., 2011] describes about new Perfect Hashing and Pruning algorithm. It scans the database only one time to mine the association rules based on hashing technique and a perfect hash function is used here with minimal collision. The main idea is to divide the hash table iteratively to store k-itemset[8]. For each part of the table, nodes and valid bit array are maintained. Nodes are used to store the number of occurrence of an item. Valid bit '1' in the array represents its frequency and '0' bit represents that item is infrequent. All non-frequent nodes in each part of hash table are deleted and the items with valid bit one in the array is called frequent items.

[Holt and Chung, 2002] presents a new technique for mining association rules between words in text database. The traditional mining algorithms cannot handle the text database effectively, because of the large number of words that need to be counted. So the multipass with Inverted Hashing and Pruning [MIHP] method is proposed [9] to mine relevant patterns that exist in database. Experimental results show that the MIHP algorithm performs better for large database.

[Huiying Wang, 2011] proposed an improved association rule mining algorithm based on reducing the time of scanning candidate itemset. It uses a hash tree to store candidate itemset. Two theorems are proposed in order to reduce the candidate items and database size. Candidate items are then stored in a hash tree and every node of the tree includes an itemset or a hash table. Every hash table points to another node. Then frequent items are mined by searching on the hash tree. The major drawback of this technique is a large memory space is utilized for storing hash tree.

[Judy C.R.Tseng et *al.*, 2006] proposed a method for mining association rules based on minimal perfect hashing scheme. Mining of association rules[20] from large and frequently updated database is one of the most important issues in data mining. By generating non-collision hashing tables, the proposed technique is suitable for handling large database containing huge amount of transaction and frequently updated data.

APRIORI ALGORITHM

The Apriori algorithm[2] is the well-known association rule mining algorithm and is used in most commercial products. Apriori algorithm is designed to operate on database containing the transactions. Apriori uses a "bottom up" approach, where frequent subsets are extended one item at a time (a step known as candidate generation), and groups of candidates are tested against the data. The algorithm terminates when no further successful extensions are found.

Apriori uses breadth-first search technique and a tree structure to count candidate itemsets efficiently. The basic idea of the Apriori algorithm is to generate candidate itemsets of a particular size and then scans the database to count whether they are large. Let Li denote the collection of large itemsets with 'i' number of items. The algorithm begins by identifying candidate 1-itemsets (C_1). Each item that has the necessary support forms the frequent 1-itemset and included in L_1 and other itemsets are dropped from consideration. This process of retaining necessary itemsets is called "pruning".

The collection C_{i+1} can be constructed by considering each pair of sets in L_1. L_{i+1} are generated by comparing the support count of C_{i+1} itemset with minimum support and store the itemsets which satisfy necessary support. This procedure is continued until all frequent itemsets up to the desired maximum size have been obtained or no further pruning is possible. Table 1 is an example transactional database. Transaction Identifier (TID) is the list of transactions for a database and it contains the items I1, I2, I3, I4 and I5. The number of candidate itemset generation is high in Apriori, therefore it takes repeated scaning of the database to find the frequent items. Figure 1 explains the steps that are followed in Apriori Algorithm.

Table no1 Example Transactional Database

TID	*List of item ID's*
T100	***I1,I2,I5***
T200	***I2,I4***
T300	***I2,I3***
T400	***I1,I2,I4***
T500	***I1,I3***
T600	***I2,I3***
T700	***I1,I3***
T800	***I1,I2,I3,I5***
T900	***I1,I2,I3***

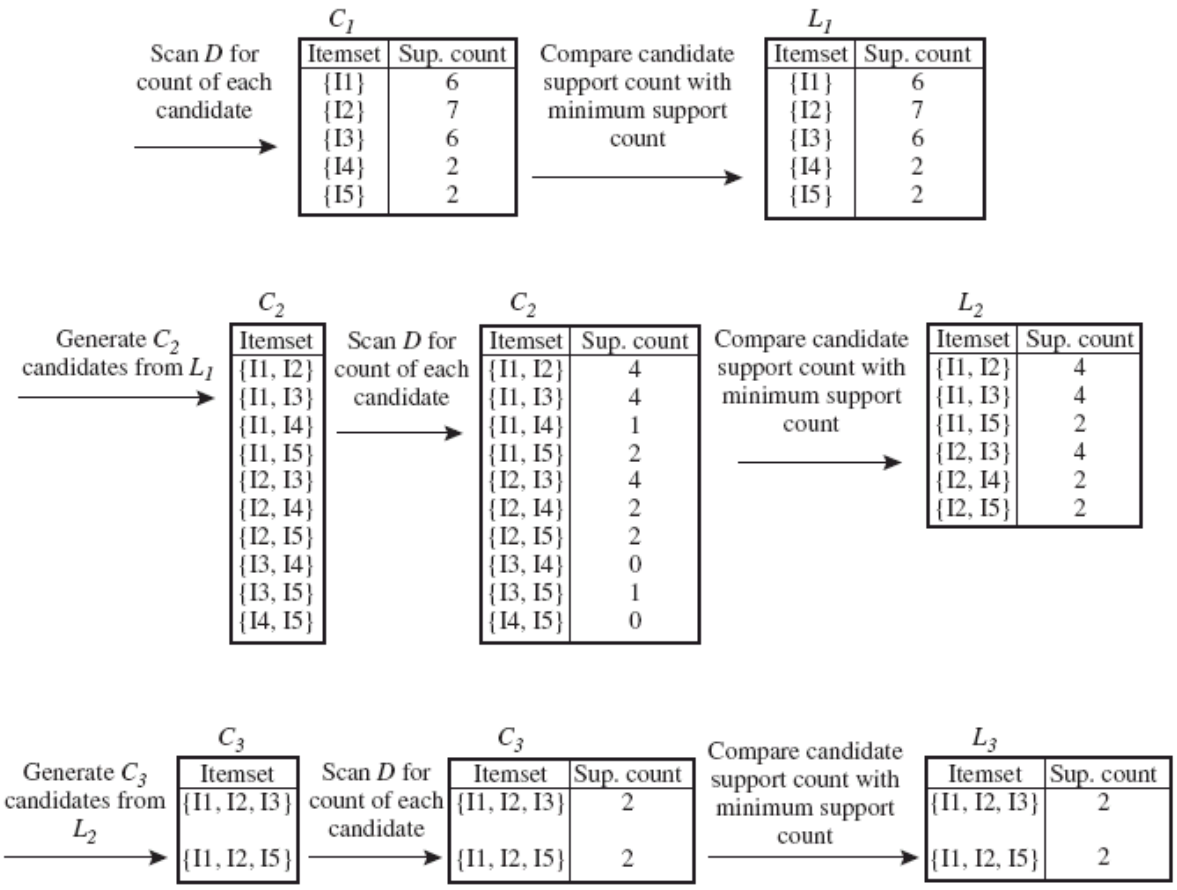


Figure No 1 Apriori Algorithm

[Minimum support = 2]

Apriori suffers from two shortcomings; first, it requires a large number of database scans. Second, a huge number of candidates are generated and their occurrence frequencies tested against the database.

H-BIT ARRAY HASHING (H-BAH) ALGORITHM

H-BAH algorithm which is a hash-based technique mines the frequent itemsets without any collision in the hash table. The lengthy probing sequence and secondary clustering that exist with

Quadratic Probing technique [12] is avoided by using an H-Bit array in header bucket of the hash table. H-Bit array technique also shrinks the size of hash table, which is required for placing itemsets. The initial transactional database as shown in table no 2 is in horizontal format and it is converted into vertical format (Itemset, Tidset).

Table No 2 Horizontal Transactional Database

Tidset	*Itemset*
T1	*I1,I2,I3,I4*
T2	*I2,I4*
T3	*I1,I5*
T4	*I2,I3*
T5	*I1,I4*
T6	*I2,I5*
T7	*I1,I3*
T8	*I3,I4*
T9	*I1,I2*
T10	*I3,I4*

Table No 3 Vertical Format of the Initial Transactional Database

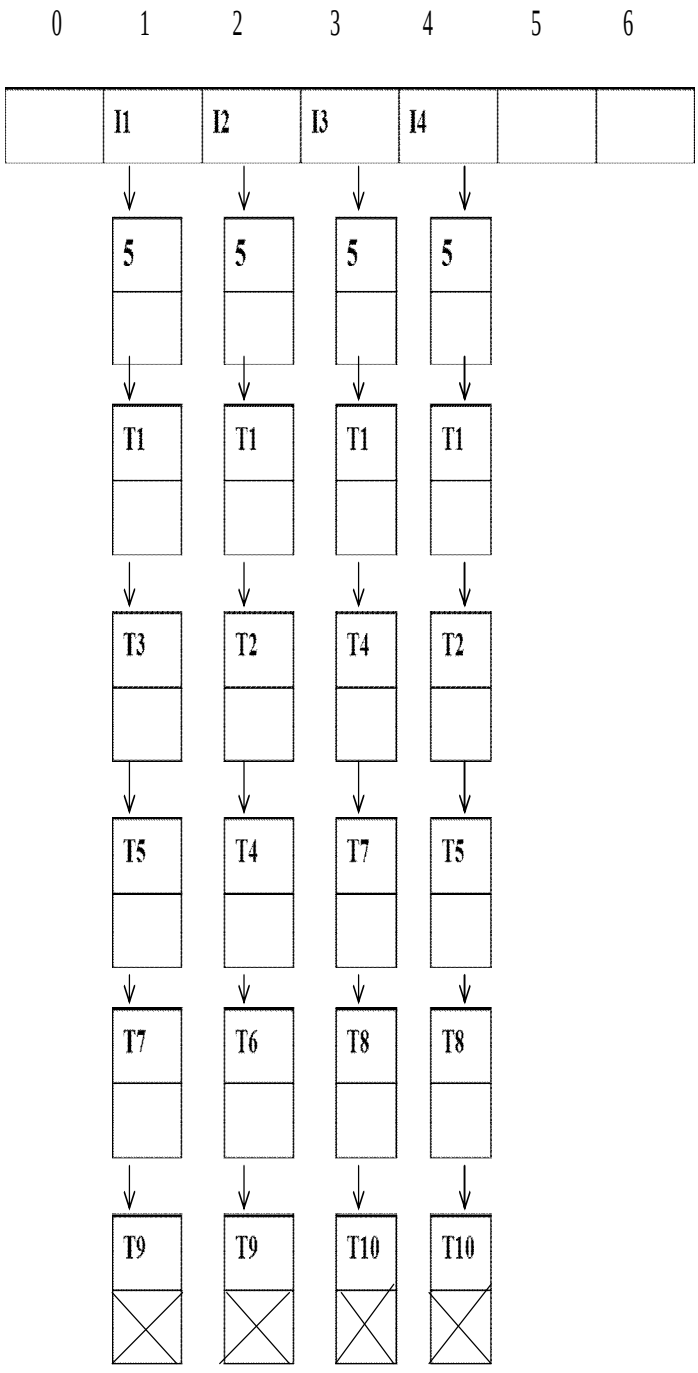
Itemset	*Tidset*
I1	*T1,T3, T5,T7,T9*
I2	*T1,T2,T4,T6,T9*
I3	*T1 ,T4,T7,T8,T10*
I4	*T1,T2,T5,T8,T10*
I5	*T3,T6*

H-Bit array technique constructs hash table based on number of items in the database and it is defined by the following function.

$$n = number\ of\ items + c$$

Candidate 1-itemsets in vertical format are hashed by using hash function. After placing all the items in the hash table, an H-Bit array is added to the first bucket or slot of the hash table to store the information of all buckets to store the items without any collision. The itemsets are placed and linked list [6] is created for every stored item in the hash table. Linked list comprised of a series of nodes and each node has data item and a pointer to next node. The first node in the list indicates support count of the item and the remaining items are placed in remaining nodes.

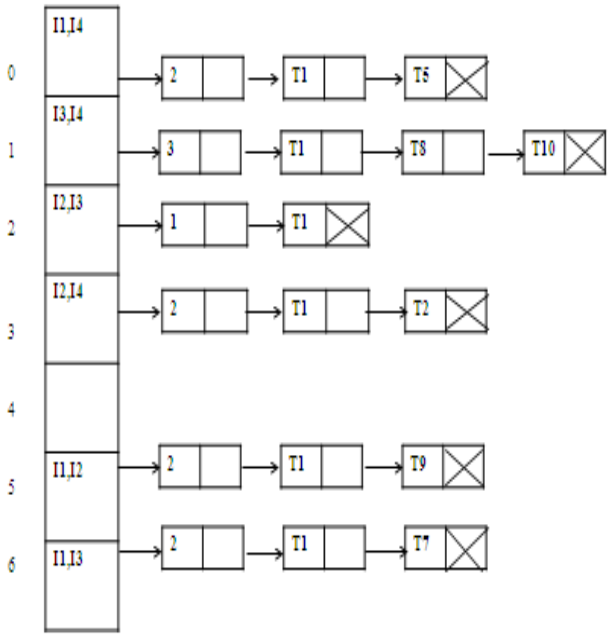
Figure No 2 Hash Table for Frequent 1-itemsets



By using the minimum support threshold frequent 1- itemsets are derived from candidate 1-itemset. If any collision occurs while hashing then items, the H-Bit array is searched for finding the hash bucket or slot which is free for placing the items.

After placing all the items in the table, its corresponding linked list is created. Then combine all possible combinations of frequent 1-items to find candidate 2-itemset.

Figure No 3 Hash Table Containing Linked List for Candidate 2-itemsets.



The same steps are repeated until all level of frequent itemset is generated.

CONCLUSION

Hashing process is done for every level and it will terminate when there is no more candidate itemsets exists. Hashing technique is mainly used for its reusability. The entries in first level hash table are deleted and same locations are used to map second level itemsets and so on. By using hash table with the linked list structures, frequent itemsets is filtered efficiently. Thus, the H-Bit array technique reduces hash table size and maps the items into hash table without any collision in a very short span of time. It also avoids lengthy probing sequence and so the secondary clustering is avoided.

REFERENCES

1. Usha. M, “Speech Therapy Models for Disabled Children Using Data Mining Techniques - A Review”, International Journal on Computer Science Trends and Technology Volume 4, Issue 3. ISSN : 2347-8578. Impact factor – 1.1.
2. R. Agarwal, T. Imielinski, and A. Swami, “Mining Association Rules Between Sets of Items in Large Databases”, In the proceedings of the ACM SIGMOD International Conference on Management of Data. 1993; 207-216.

3. R. Agarwal and Srikant. R, "Fast Algorithms for Mining Association Rules", In the proceedings of 20th International Conference on Very Large Data Bases. 1994; 487-499
4. Chin-Chen Chang and Chin-Yang Lin, "Perfect Hashing Schemes for Mining Association Rules", The Computer Journal, 2005; 48: 168-179.
5. Ebrahim Ansari Chelche, M. H. Sadreddini and G. H. Dastghaybifard, "Using Candidate Hashing and Transaction Trimming in Distributed Frequent Itemset Mining", World Applied Sciences Journal. 2010; 9(12) : 1353-1358.
6. NVB Gangadhara Rao and Sirisha Aguru, "A Hash based Mining Algorithm for Maximal Frequent Item Sets using Double Hashing", Journal of Advances in Computational Research, 2012; 1 : 1-2
7. Hannu Toivonen, "Sampling Large Databases for Association Rules", Proceedings of the 22nd International Conference on Very Large Data Bases. 1996; 134-145 (ISBN: 55860-382-4)
8. J.Han and M.Kamber, "Data Mining: Concepts and Techniques", second edition, Morgan Kaufmann Publishers, 2006 (ISBN: 1-55860-901-6).
9. Hassan Najadat, Amani Shatwani and Ghadeer Objedat, "A New Perfect Hashing and Pruning Algorithm for Mining Association Rule", IBIMA Publishing, Article 2011 Id: 652178.
10. John D.Holt and Soon M.Chung, "Mining Association Rules using Inverted Hashing and Pruning", ELSEVIER Information Processing letters. 2002; 211-220
11. Jochen Hipp, Ulrich Guntzer, and Gholamreza Nakhaeizadeh, "Algorithms for Association Rule Mining – A General Survey and comparison", ACM SIGKDD Explorations, 2000; 2 (1): 58-64
12. M.Krishnamurthy, A.Kannan, R.Baskaran and R.Deepalakshmi,"Frequent Itemset Generation using Hashing-Quadratic Probing Technique", European Journal of Scientific Research, 2011; 50(4): 523-532
13. SA Ozel and HA Guvenir, "An Algorithm for Mining Association Rules using Perfect Hashing and Database Pruning", In the 10th Turkish Symposium on Artificial Intelligence, 2001
14. J.S.Park and M.Chen, "An Effective Hash Based Algorithm for Mining Association Rules", ACM SIGMOD International Conference on Management of Data, 1995
15. Jang Soo Park, Ming-Syan Chen and Philip S.Yu, "Using a Hash-Based Method with Transaction Trimming for Mining Association Rules", IEEE Transactions on Knowledge and Data Engineering, 1997; 9(5).

16. Ramakrishnan Srikant, Quoc Vu and Rakesh Agarwal, "Mining Association Rules with Item Constraints", In the proceedings of 3rd International Conference on Knowledge Discovery and Data Mining, 1997
17. Ratchadaporn Amornchewin, "Probability-Based Incremental Association Rules Discovery Algorithm with Hashing Technique", International Journal of Machine Learning and Computing (IJMLC). 2011; 1: 1, (ISSN: 2010-3700).
18. R. Rathinasabapathy, R.Bhaskaran, "Analysis of Complexities for Finding Efficient Association Rule Mining Algorithm", International Journal of Internet Computing (IJIC), 2011
19. T Shintani and M Kitsuregawa, "Hash Based Parallel Algorithms for Mining Association Rules", In the proceedings of 4th International Conference on Parallel and Distributed Information Systems. 1996; 19-30.
20. Judy C. R. Tseng, Gwo-Jen Hwang and Wen-Fu Tsai, "A Minimal Perfect Hashing Scheme to Mining Association Rules from Frequently Updated Data", Journal of the Chinese Institute of Engineers, 2006; 29: 3.
21. K. Vanitha and R. Santhi, "Using Hash Based Apriori Algorithm to Reduce the Candidate 2-itemsets for Mining Association Rule", Journal of Global Research in Computer Science, 2011; 2:5 (ISSN-2229-371-X).
22. H Wang and X Liu, "The Research of Improved Association Rules Mining Apriori Algorithm", In the 8th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD). 2011; 2: 961-964.