International Journal of Contemporary Research in Computer Science and Technology (IJCRCST) ISSN: 2395-5325
Volume 3, Special Issue 3 (September '2017)
Proceeding of - International Conference on Recent Trends in Computational Life Science and Information Technology,
Conference held at Tiruppur Kumaran College for Women,Tirupur,Tamilnadu, India.

ASSOCIATION RULE MINING USING PERFECT HASHING AND PRUNING (PHP) ALGORITHM

M.Preethi,
M.Phil.,Computer Science),
Kongu Arts and Science College,
Erode, Tamilandu, India.

P.K.Mangaiyarkarasi,
Associate Professor,
Kongu Arts and Science College,
Erode, Tamilandu, India.

Abstract: Knowledge mining means extracting the knowledge from the database. Software algorithms are implemented for extracting the knowledge from database. In data mining each algorithm has a different objective and to obtain meaningful and previously unknown patterns from large dataset is an emerging and challenging problem. The capacity of the hardware architecture is fixed. As number of candidate itemsets or the number of items in the database is larger than the hardware capacity. So That the items are loaded into the hardware separately, Due to this time complexity is more to load candidate itemsets or database items into the hardware is in proportion to the number of candidate itemsets multiplied by the number of items .in the database. Increase of candidate itemsets and a large database would create a performance blockage. We propose a Hash-based and Pipelined (HAPPI) architecture for hardware-enhanced association rule mining. In HAPPI architecture, we propose PHP algorithm.

Keywords: Knowledge mining, Hash-based, Pipeline, PHP.

I.INTRODUCTION

Data Mining can be used to provide useful information from the database. As the size of the database is upgraded, there is a need to implement a new architecture to reduce the size of the candidate item sets. Association Rule Mining was introduced by it has emerged as a prominent research area. The Association Rule Mining problem also referred to as the market basket problem can be formally defined as follows.

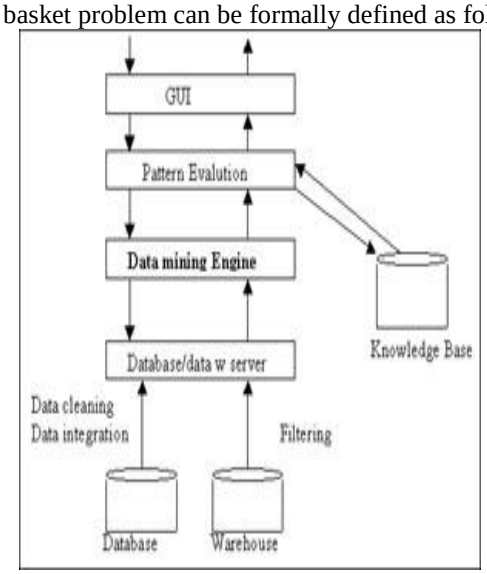

Figure1: Association rule mining process

Let $I = \{i_1, i_2, \ldots\ldots\ldots, i_n\}$ be a set of items and $S = \{s_1, s_2, \ldots\ldots\ldots, s_m\}$ be a set of transactions, where each transactions $s_i \in S$ is a set of items that is $s_i \subseteq I$. An Association Rule denoted by $X \Rightarrow Y$. Several measures have been introduced to define the strength of the relationship between item sets X and Y such as support, confidence and interest. The definitions of these measures are given below:

Support $(X \Rightarrow Y) = P(X, Y)$ or the percentage of transactions in the database that contain both *X* and *Y*.

Confidence $(X \Rightarrow Y) = P(X, Y) \mid P(X)$, or the percentage of transactions containing *Y* in transactions those contain *X*.

Interest $(X \Rightarrow Y) = P(X, Y) \mid P(X)\,P(Y)$, represents a test of statistical independence.

II.BOOLEAN ASSOCIATION MINING

Given a set of items I = {i1, i2……., in}, a transaction t is defined as a subset of items such that t□2I, where 2I = {□, {i1}, {i2}, …, {in}, {i1, i2}, …, { i1, i2, …, in}}. In reality, not all possible transactions might occur. For example, transaction t = □ is excluded. Let T □ 2I be a given set of transactions {t1, t2, …, tm}. Every transaction t□T has an assigned weight w'(t). Several possible weights could be considered, w'(t) = 1, for all transactions t□T. w'(t) = f(t), where f(t) is the frequency of transaction t, for all transactions t□T, i.e., how many times the transaction t was repeated in our database. w'(t) = |t| * g(t) for all transactions t□T, where |t| is the cardinality of t, and g(t) could be either

one of the weight functions w'(t)'s defined in (i) and (ii). In this case, longer transactions get higher weight. w'(t)= v(t) * f(t) for all transactions t□T, where v(t) could be the sum of the prices or profits of those items in t. The weights w's are normalized to

$$w(t) = \frac{w'(t)}{\sum_{\forall\, t' \in T} w'(t')}, \text{ and } \sum_{\forall\, t \in T} w(t) = 1$$

IJCRCST © 2017 | All Rights Reserved

International Journal of Contemporary Research in Computer Science and Technology (IJCRCST) ISSN: 2395-5325
Volume 3, Special Issue 3 (September '2017)
Proceeding of - International Conference on Recent Trends in Computational Life Science and Information Technology,
Conference held at Tiruppur Kumaran College for Women,Tirupur,Tamilnadu, India.

III.THE HAPPI ARCHITECTURE

In HAPPI architecture, there are three hardware modules. First, when the database is fed into the hardware, the candidate itemsets are compared with the items in the database by the systolic array. Candidate itemsets that have a higher frequency than the minimum support value are viewed as frequent itemsets. Second, we determine the frequency that each item occurs in the candidate itemsets in the transactions at the same time. These frequencies are called trimming information. From this information, infrequent items in the transactions can be eliminated since they are not useful in generating frequent itemsets through the trimming filter. Third, we generate item sets from transactions and hash them into the hash table, which is then used to filter out unnecessary candidate itemsets. After the hardware compares candidate itemsets with the items in the database, the trimming information is collected and the hash table is built. The useful information helps us to reduce the number of items in the database and the number of candidate itemsets. Based on the trimming information, items are trimmed if their corresponding occurrence frequencies are not larger than the length of the current candidate itemsets. In addition, after the candidate itemsets are generated by merging frequent subitemsets, they are sent to the hash table filter. If the number of itemsets in the corresponding bucket of the hash table is less than the minimum support, the candidate itemsets are pruned. As such, HAPPI solves the bottleneck problem mentioned earlier by the cooperation of these three hardware modules. To achieve these goals, we devise the following five procedures in the HAPPI architecture: support counting, transaction trimming, hash table building, candidate generation, and candidate pruning. Moreover, we derive several formulas to decide the optimal design in order to reduce the overhead induced by the pipeline scheme and the ideal number of hardware modules to achieve the best utilization.

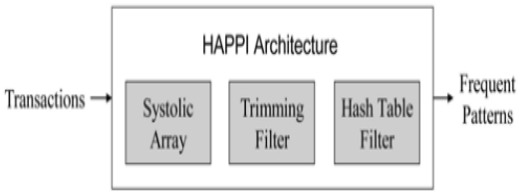

Figure2: The HAPPI architecture

The HAPPI architecture consists of a systolic array, a trimming filter, and a hash table filter. There are several hardware cells in the systolic array. Each cell can perform the comparison operation. Based on the comparison results, the cells update the support counters of candidate itemsets and the occurrence frequencies of items in the trimming information. A trimming filter then removes infrequent items in the transactions according to the trimming information. In addition, we build a hash table by hashing itemsets generated by each transaction. The hash table filter then prunes unsuitable candidate itemsets.

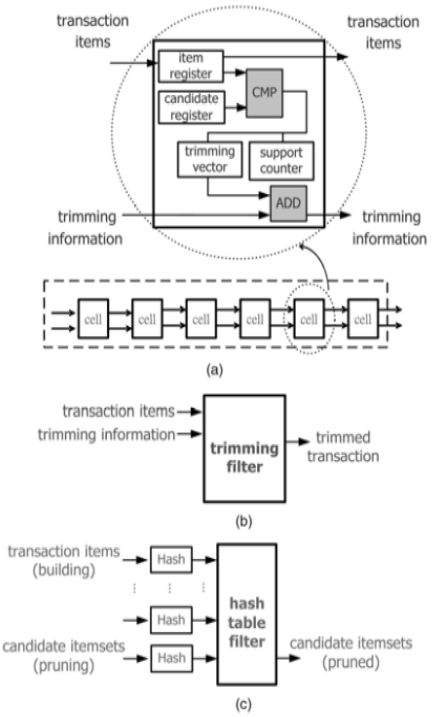

Figure3: HAPPI architecture: a) Systolic array, 2) Trimming filter and 3) Hash table filter

IV.DIRECT HASHING WITH EFFICIENT PRUNING FOR DATA MINING (DHP) ALGORITHM

Direct hashing and pruning(DHP) varies from the apriori algorithm in the sense that it uses hashing technique to remove the unwanted itemsets for the generation of next set of candidate itemsets. In apriori algorithm, determining all the large itemsets from the set of candidate large itemsets for a particular pass by scanning the database become very expensive. But in DHP the size of the candidate itemsets is reduced and then it performs much faster than the apriori algorithm.

V.PERFECT HASHING AND PRUNING(PHP) ALGORITHM

In DHP it generates a number of false positives and so perfect hashing and pruning (PHP) is another algorithm suggested as an improvement over DHP. In this algorithm, during the first pass a hash table with size equal to the number of individual items in the database is created. Then each individual item is mapped to different item in the hash table. After the first pass the hash table consists of exact number of occurrences of each item in the database. Then the algorithm removes all the transactions which have no

items that are frequent and also trims the items that are not frequent. Simultaneously, candidate k-itemsts are generated and the occurrences of each are found out. Thus at the completion of each pass, the algorithm gives a pruned database, occurrences of candidate k-itemsets and set of frequent k-itemsets and algorithm terminates when no new frequent k-itemsets are found. Thus PHP is an improvement over DHP with n false positives and thus extra time required

IJCRCST © 2017 | All Rights Reserved

for counting frequency of each item set is eliminated. PHP also reduces the size of the database.

VI.CONCLUSION

Data in this research we conclude that with the help of hash based pipelining technique products in market can be sold faster because in HAPPI technique it removes bottleneck problem thereby providing faster throughput and our sales process becomes faster because due to indexing hashing process becomes faster. Firstly items are kept in systolic array then items which are not in close proximity with each other are trimmed or removed from the filter then put into hash table filter so that duplication of items get removed so in this way. It solves our bottleneck problem.

VII.REFERENCES

[1] R. Agarwal, C. Aggarwal, and V. Prasad, "A Tree Projection Algorithm for Generation of Frequent Itemsets," *J. Parallel and Distributed Computing*, 2000.

[2] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," *Proc. 20th Int'l Conf. Very Large Databases(VLDB)*, 1994.

[3] Z.K. Baker and V.K. Prasanna, "Efficient Hardware Data Mining with the Apriori Algorithm on FPGAS," *Proc. 13th Ann. IEEE Symp. Field- Programmable Custom Computing Machines (FCCM)*, 2005.

[4] S. Cong, J. Han, J. Hoeflinger, and D. Padua, "A Sampling-Based Framework for Parallel Data Mining," *Proc. 10th ACM SIGPLAN Symp. Principles and Practice of Parallel Programming (PPoPP '05)*, June 2005.

[5] Anthony K.H. Tung, Hongjun Lu, Jiawei Han, Member, and Ling Feng, "Efficient Mining of Inter transaction Association Rules" *IEEE Transactions on Knowledge and Data Engineering*, Vol. 15, No.1,January/February'03.

[6] Cheung D, V.T Ng, A. Fu, and Y.Fu. "Efficient mining of association rules in distributed databases". *IEEE Trans. Knowledge and Data Engineering*, pp 1-23, 1996.