Volume : 1, Issue : 1
January - June 2011

ISSN : 2229 - 3515

international journal of ADVANCES IN SOFT COMPUTING TECHNOLOGY

Editor-in-Chief
Dr.Vaka Murali Mohan

Published by
BHAVANA RESEARCH CENTER

HAPPI Architecture by using Mining Association Rules

***Pratiba Reddy, G*[1*]*., Tulasi Kalyani, P*[2] *and Kumar, D*[3]**

1. St. Marys College of Engineering &Technology, Deshmukhi, Hyderabad, AP, INDIA
2. Joginaplly Bhaskara Institute of Technology, Moinabad, Hyderabad, AP, INDIA
3. P.IndraReddy(M) Engineering College, Chevella, R.R(Dt), AP, INDIA

KeyWords:
HAPPI Architecture, Association Rule, Hashing, Trimming, Pipeling

Abstract: *The HAPPI architecture for hardware enhancement by using association rule mining and incorporate the pipeline methodology is presented in this paper. The holdup of a hardware schemes is related to the number of candidate item sets and the size of the database. HAPPI effectively solves the holdup problem. The HAPPI architecture is explained in three sections the first one is system architecture, second one is trimming filter, and third one is hash table filter.*

1. Introduction:

Association rules are well recognized as a data mining tool for analysis of transactional data. Wide spectrums of methods for mining associations have been proposed up to date, including batch and incremental approaches. Most of the accurate incremental methods minimize, but do not completely eliminate reruns through processed data. Due to the increasing use of very large databases and data warehouses, mining useful information and helpful knowledge from transactions is evolving into an important research area. Most conventional data-mining algorithms identify the relationships among transactions using binary values and find rules at a single concept level. Transactions with quantitative values and items with hierarchy relation are, however, commonly seen in real-world applications.

Association rule mining have been applied to various datasets, due to their practical usefulness. Little attention has been paid, however, on how to apply the association mining techniques to analyze questionnaire data. Categorical data can generally be classified into ordinal data and nominal data.

* **Mrs. Pratiba Reddy, G**
Associate Professor
Department of Computer Science & Engineering
St. Marys College of Engineering &Technology,
Deshmukhi, Hyderabad, AP, INDIA
Ph. No.: 91-9966483759
E-mail: nandini19_chilka@rediffmail.com

Although there have been numerous studies on finding association rules from nominal data, few have tried to do so from ordinal data. Additionally, previous mining algorithms usually assume that the input data is precise and clean, which is unrealistic in practical situations. Real-world data tends to be imprecise due to human errors, instrument errors, recording errors, and so on. Earlier lot of research was done by several researchers some of them are Yen-Liang Chen and Cheng-Hsiung Weng [1] developed a unified approach based on fuzzy techniques so that all different data types can be handled in a uniform manner. Xiang-Rong Jiang and Le Gruenwald [2] used the association rules to mine the association relationships among different genes under the same experimental conditions. Tzung-Pei Hong et al [3] proposed a fuzzy mining algorithm for extracting implicit generalized knowledge from transactions stored as quantitative values. Damian Dudek [4] proposed a new approximate algorithm RMAIN for incremental maintenance of association rules. RMAIN is fully separated from a rule mining algorithm and this independence makes it highly general and flexible. Moreover, it operates on rules in their final form, ready for decision support, and not on intermediate representation (frequent itemsets), which requires further processing. These features make the RMAIN algorithm well suited for rule maintenance within knowledge bases of autonomous systems with strongly bounded resources and time for decision making. Tutut Herawan and Mustafa Mat Deris [5] presented an alternative approach for mining regular association rules and maximal association rules from transactional datasets using soft set theory. Yen-Liang Chen and Cheng-Hsiung Weng [6] proposed a new approach to discovering association rules from imprecise ordinal data.

The HAPPI architecture by using association rules consists of support counting, transaction trimming, hash table building, candidate generation, and candidate pruning. Moreover, several formulas were derived 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. The execution time between sequential processing and pipeline processing is also analyzed in this paper. Several experiments were conducted to evaluate the performance of the HAPPI architecture. The performance of HAPPI is better than that of the previous approach when the systolic array contains different numbers of hardware cells. The advantages of the HAPPI architecture are that it has more computing power and saves the space costs for mining association rules in hardware design. The scale-up experiment also shows that HAPPI outperforms the previous approach on different numbers of transactions in the database. Indeed, our architecture is a good example to demonstrate the methodology of performance enhancement by hardware. The performance of HAPPI will further be improved. In view of the fast increase in the amount of data in various emerging mining applications (e.g., network application mining, data stream mining, and bioinformatics data mining), it is envisaged that hardware enhanced mining is an important research direction to explore for future data mining tasks.

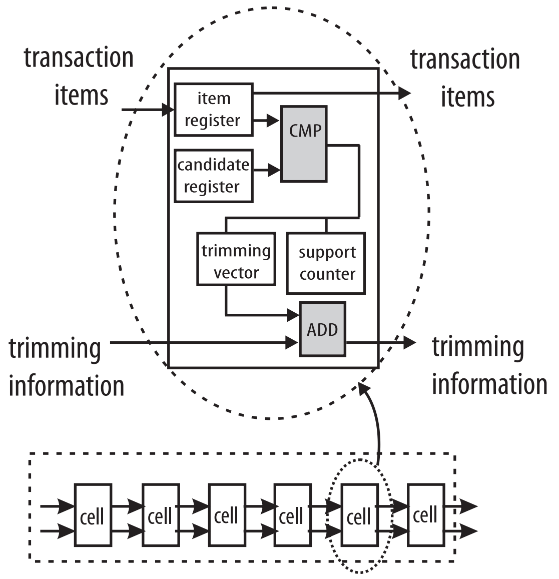

Fig.1a. HAPPI System Architecture

2. HAPPI ARCHITECTURE

Hardware schemes have to load candidate item sets' and the database into the hardware to execute the comparison process. Large number of data sets and database would cause a performance holdup. To overcome this problem, the HAPPI architecture is proposed to deal with efficient hardware-enhanced association rule mining and incorporate the pipeline methodology into the HAPPI architecture to perform pattern matching and collect useful information to reduce the number of candidate, data sets and database all together. Like this HAPPI effectively solves the holdup problem. The HAPPI architecture shown in figure 1(a,b,c) as (a) system architecture (b) trimming filter, and (c) hash table filter.

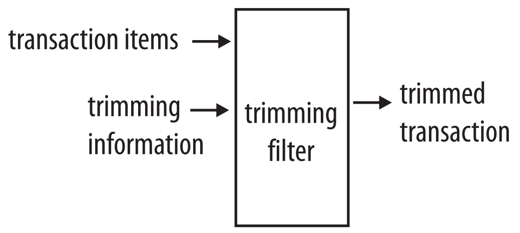

Fig.1b. Trimming Filter

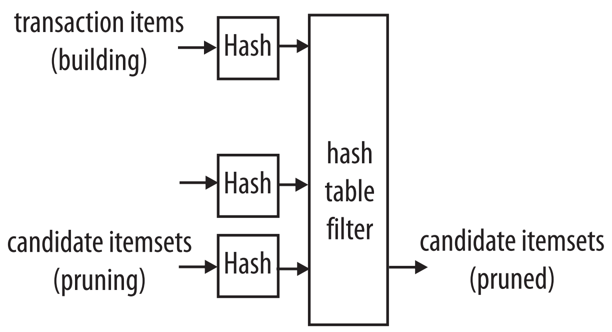
transaction items (building)
Hash
Hash
candidate itemsets (pruning)
Hash
hash table filter
candidate itemsets (pruned)

Fig.1c. Hash Table Filter

2.1 System Architecture

The HAPPI system architecture is shown in figure 1(a) and consists of a systolic array, a trimming filter, and 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.

Pipeline Design:

The transaction trimming procedure has to obtain trimming information to execute the trimming process. However, this process cannot be completed until the support counting procedure compares all the transactions with all the candidate data sets. In addition, the hash table building procedure has to get the trimmed transactions from the trimming filter after all the transactions have been trimmed. This problem can be resolved by applying the pipeline scheme, which utilizes the three hardware modules simultaneously in the HAPPI framework. First, we divide the database into Npipe parts. One part of the transactions in the database is streamed into the systolic array and the support counting process is performed on all candidate data sets. After comparing the transactions with all the candidate data sets, the transactions and their trimming information are passed to the trimming filter first. The systolic array then processes the next group of transactions. After items have been trimmed from a transaction by the trimming filter, the transaction is passed to the hash table filter, as shown in Fig. 2, and the trimming filter can deal with the next transaction. In this way, all the hardware modules can be utilized simultaneously. Although the pipelined architecture improves the system's performance, it increases the computational overhead because of multiple times of loading candidate data sets into the systolic array.

2.2 Transaction Trimming

In the HAPPI architecture, the trimming information records the frequency of each item in a transaction that appears in the candidate data sets. The support counting and trimming information collecting operations are similar since they· all need to compare candidate data sets with transactions. Therefore, in addition to transactions in the database, their corresponding trimming information is also fed into the systolic array in another pipe, while the support counting process is being executed. Transaction trimming is divided into two sections the first one is the Stream a transaction into the cell and it is shown in Figure 3a; the second one is the stream trimming information into the cell and it is shown in Figure3b.

2.3 Hash Table Filtering

A hash value generator and hash table updating module is used to build a hardware hash table filter. The former generates all the k-itemset combinations of the transactions and puts the k-itemsets into the hash function to create the corresponding hash values. As shown in figure 4, the hash value generator comprises a transaction memory, a state machine, an index array, and a hash function. The transaction memory stores all the items of a transaction. The state machine is the controller that generates control signals of different lengths (k=2.3…) flexibly. Then, the control signals are fed into the index array. To generate a k-itemset, the first k entries in the index array are utilized.

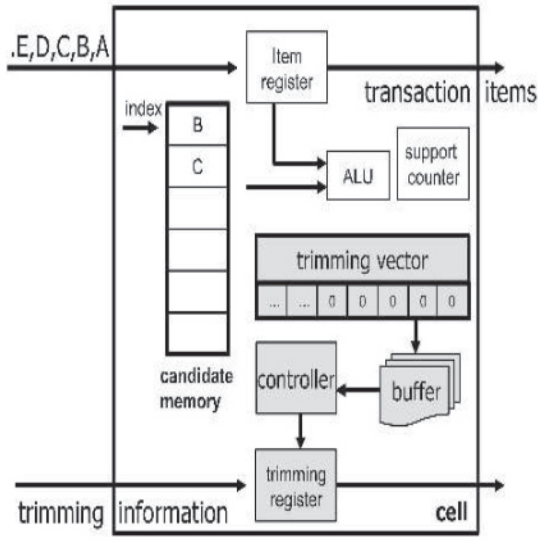

Fig.3a. Stream a transaction into the cell

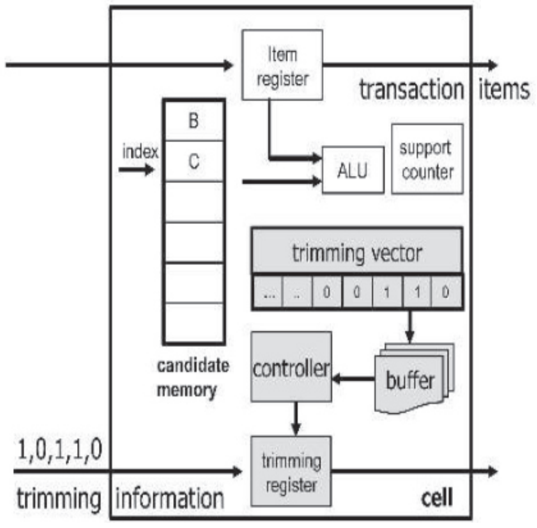

Fig.3b. Stream trimming information into the cell

The values in the index array are the indices of the transaction memory. The item selected by the ith entry of the index array is the i th item in a k-itemset. By changing the values in the index array, the state machine can generate different combinations of k-itemsets from the transaction. The procedure starts by loading a transaction into the transaction memory. Then, the values in the index array are reset, and the state machine starts to generate control signals. The values in the index array are changed by the different states. Each item in. the generated itemset is passed to the hash function through the multiplexer. The hash function takes some bits from the incoming k-itemsets to calculate the hash values.

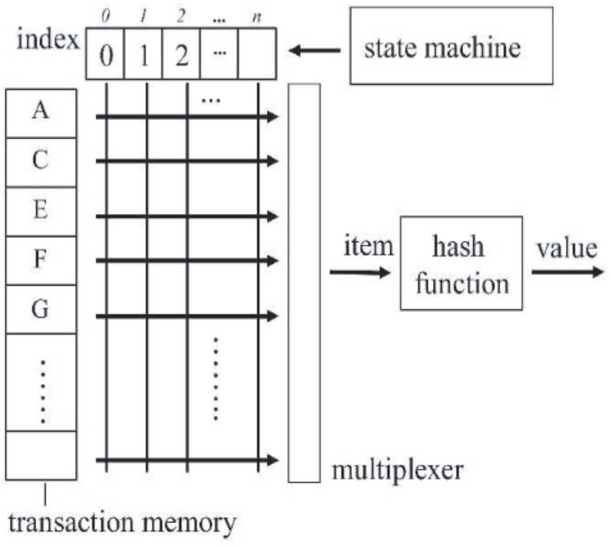

Figure 4. The hash value generator

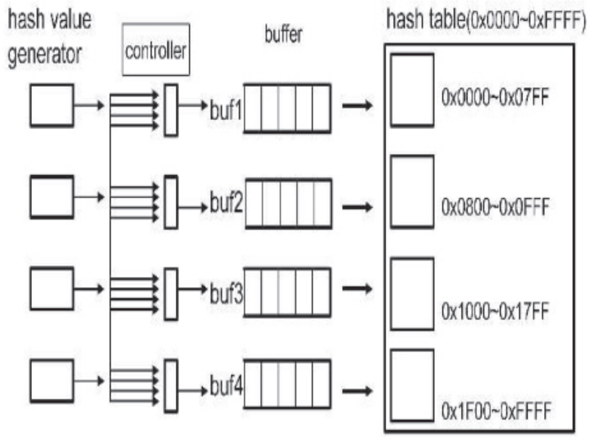

Figure 5 The parallel hash table building module

The hash values generated by the hash value generator are passed to the hash table updating module. To speed up the process of hash table building, we utilize Nparallel hash value generators so that the hash values can be generated simultaneously. In addition, the hash table is divided into several parts to increase the throughput of hash table building. Each part of the hash table contains a range of hash values, and the controller passes the incoming hash values to the buffer they belong to. These hash values are taken as indexes of the hash table to accumulate the values in the table, as shown in Figure 5.

3. CONCLUSION

The HAPPI architecture for hardware enhancement by using association rule mining and incorporate the pipeline methodology is presented in this paper. The holdup of a hardware schemes is related to the number of candidate item sets and the size of the database. HAPPI effectively solves the holdup problem. The HAPPI architecture is explained in three sections the first one is system architecture, second one is trimming filter, and third one is hash table filter.

4. REFERENCES

1. Yen-Liang Chen and Cheng-Hsiung Weng "Mining fuzzy association rules from questionnaire data" Knowledge-Based Systems, Vol. 22, Issue 1, January 2009, pp 46-56.
2. Xiang-Rong Jiang and Le Gruenwald "Microarray gene expression data association rules mining based on BSC-tree and FIS-tree" Data & Knowledge Engineering, Vol. 53, Issue 1, April 2005, pp 23-29.
3. Tzung-Pei Hong., Kuei-Ying Lin and Shyue-Liang Wang "Fuzzy data mining for interesting generalized association rules" Fuzzy Sets and Systems, Vol. 138, Issue 2, 1 Sept 2003, pp255-269
4. Damian Dudek "RMAIN: Association rules maintenance without reruns through data" Information Sciences, Vol. 179, Issue 24, 15 Dec 2009, pp 4123-4139.
5. Tutut Herawan and Mustafa Mat Deris "A soft set approach for association rules mining" Knowledge-Based Systems, Vol. 24, Issue 1, Feb 2011, pp 186-195.
6. Yen-Liang Chen and Cheng-Hsiung Weng "Mining association rules from imprecise ordinal data" Fuzzy Sets and Systems, Vol. 159, Issue 4, 16 Feb 2008, pp 460-474.