ISSN 2229-5518

Association Rule Mining based on Apriori Algorithm in Minimizing Candidate Generation

Sheila A. Abaya

Abstract— Association Rule Mining is an area of data mining that focuses on pruning candidate keys. An Apriori algorithm is the most commonly used Association Rule Mining. This algorithm somehow has limitation and thus, giving the opportunity to do this research. This paper introduces a new way in which the Apriori algorithm can be improved. The modified algorithm introduces factors such as set size and set size frequency which in turn are being used to eliminate non significant candidate keys. With the use of these factors, the modified algorithm introduces a more efficient and effective way of minimizing candidate keys.

Index Terms— Apriori algorithm, data mining, frequent items, set size

1 Introduction

DATA mining is an important research domain nowadays that focuses on knowledge discovery in databases. It is where data from the database are mined so that informative data can be generated and used effectively and efficiently by humans. Its objective is prediction and description [10]. One of the aspects of data mining is the Association Rule mining. It consists of two procedures: [10] First, finding the frequent itemset in the database using a minimum support and constructing the association rule from the frequent itemset with specified confidence. It relates to the association of items wherein for every occurrence of A, there exists an occurrence of B. This mining is more applicable in the market basket analysis [9]. That application is helpful to the customers that buy certain items. That for every item that they bought, what would be the possible item/s coupled with the purchased item. Apriori algorithm is the most widely used association rule mining algorithm [9]. However, several limitations have been discovered in this method [7] such as:

- Several iterations of data are needed for mining data
- Usually generates items which are irrelevant
- Difficulties in finding unusual events

With these limitations several works have been noted to improve the efficiency of Apriori algorithm.

2 Related Works

In the work of Mamta[7], an efficient approach is the use of weight factor and utility for mining high utility patterns. It uses several attributes in mining frequent itemset. Such attributes are: calculating the profit ratio using the q-factor equation[4],

$$Q - Factor = P / \sum Pi \qquad (1)$$

applying the Apriori algorithm with its pruning stage, frequent item selection using the confidence measure and calculating the weighting factor [6].

$$PW = \sum_{i=1}^{n} frequency * Q - Factor \qquad (2)$$

In the proposed work of S. Prakash[2],], it concerns about the quantitative association rule mining and it also deals with the reduction of memory utilization during the pruning phase. The proposed level-wise approach accesses the database more often thus results into less usage of memory. In the research of J. Hossen[5], the modified Apriori algorithm for rule formation starts with the clusters identified in the fuzzy c-means clustering method. The fuzzy rules from this work are as follows:

Rule 1:

If cluster1 in d=1 dimension^cluster1 in d=2 dimension ^ cluster1 in d=3 dimension THEN Result 1

- *The author is currently pursuing her Doctoral degree in Information Technology at Technological Institute of the Philippines, Quezon City, Philippines. E-mail: sheila_abaya@yahoo.com.ph*

Rule 2:

If cluster2 in d=1 dimension^cluster2 in d=2 dimension ^ cluster2 in d=3 dimension THEN Result 2

These rules seek to find the combination of clusters common in every cluster. More improved clustering techniques in Apriori algorithm with its rule formation can be found on the works of D. Kerana[1] mining frequent itemsets from k-1 itemsets. If k is greater than the size of the transaction T, there is no need to scan the transaction T which is generated by k-1 itemsets. The Partition Algorithm for Frequent Items (PAFI) reduces the scans of the database thereby improving the efficiency of Apriori algorithm and this is possible with the implementation of clustering method. With the work of S. Murali [11], due to the increase of data, mining frequent itemsets even in the text mining domain have been amplified. Murali also made used of cluster analysis [3], the mined frequent itemsets that were derived from meeting the defined threshold were arranged in descending order. Then, splitting the documents into partition using the resulting frequent itemsets arrived at an ensuing cluster using the derived keyword. The approach for this work follows the processes:

- Text preprocessing
- Mining of frequent itemsets
- Partitioning the text documents based on frequent itemsets
- Clustering of text documents within the partition

With E. Ramaraj[2], based on this work, the efficiency of the apriori algorithm in terms of space and time can be improved by implementing a bit stream mask search algorithms wherein the input file is converted into numerical data and the transaction file is compressed into an array where further processing is done.

3 APRIORI ALGORITHM

This algorithm follows three steps [8]:

1. For l from 1 to I do
2. For each set JI such that for each h ϵ JI occurs in at least k baskets do
3. Examine the data to determine whether the set JI occurs in at least k baskets

For this algorithm, most of its time has been spent in accessing the database until it results into one frequent association match. Based on this work, the probability model [8] was calculated with two quantities:

1. Success rate : the probability that the set is a success
2. Failure rate: the probability that the set is a failure

With this probability model, it brings out the main performance features of the algorithm.

4 MODIFIED APRIORI ALGORITHM

The improved algorithm for Apriori takes for the set size which is the number of items per transaction and set size frequency which is the number of transactions that have at least "set size" items.

Given:

Minimum Support = 3

Set size – number of items per transaction

Set size frequency – number of transactions that have at least set size" items

TABLE 1
INITIAL DATA SET WITH SET SIZE

Transaction_id	Items	*Set Size*
T1	K, A, D, B	*4*
T2	D, A, C, E, B	*5*
T3	C, A, B, E	*4*
T4	B, A, D	*3*

TABLE 2
SET SIZE FREQUENCY OF THE INITIAL DATA SET

Set Size	*Set Size Frequency*
4	3
5	1
3	4

1. Remove items with frequency less than the minimum support value.

TABLE 3
DATA SET WITH ELIMINATED FREQUENCY

Transaction_id	Items	*Set Size*
T1	A, D, B	*3*
T2	D, A, B	*3*
T3	A, B	*2*
T4	B, A, D	*3*

2. Determine initial set size to build – get the highest set size whose frequency is greater than or equal to minimum support (set size 3)

TABLE 4
SET SIZE WITH FREQUENCY MEETING MIN. SUPPORT

Set Size	Frequency
3	3
2	1

3. Get list of items from transactions with set size>= the set size determined in step #2.

TABLE 5
TRANSACTION ITEMS MEETING SET SIZE

Transaction_id	Items	*Set Size*
T1	A, D, B	*3*
T2	D, A, B	*3*
T4	B, A, D	*3*

Unique items = A, B, D

4. Create combinations with size 3 (as determined in step #2Determine the transaction numbers whose set size >= result of #1. Count the combinations' frequency in the database.

{A, B, D} = 3

5. Remove combinations with frequency less than the minimum support.

{A, B, D} = 3

6. If list of combinations in #5 is greater than 1 use the next available set size as determined in step #2. Go back to step #3.

NOTE: if the list of set sizes in step #2 is exhausted, continue moving to the next lower value until you reach set size=2.

5 IMPLEMENTATION

The original and modified algorithms were implemented using PERL. Moreover, the Benchmark module was also used to measure the execution time of the codes. This module also counts the database passes of both algorithms. Several relations have been used as test data to determine the efficiency and accuracy of both implementations.

6 RESULTS AND EVALUATION

There were 5 relations used as test data to evaluate the efficiency and accuracy of both the original apriori and the modified one.

TABLE 6
EXECUTION TIME AND DB PASSES FOR 5 DATA SET

O1	M1	Execution Time	DB Passes
0.28	0.06	79%	
343	19		94%

O2	M2	Execution Time	DB Passes
0.05	0.05	0%	
53	21		60%

O3	M3	Execution Time	DB Passes
0.01	0.01	0%	
13	15		-15%

O4	M4	Execution Time	DB Passes
0.03	0.02	33%	
13	15		-15%

O5	M5	Execution Time	DB Passes
0.05	0.01	80%	
31	18		42%

	Average	38%	33%

The average results for both the execution time and the database pass yields 38% and 33% respectively in favor of the modified one. However, in some test data the outcome is in accordance with the original algorithm. It has been observed that as the number of items per transaction decreases the favorable result will be from the original algorithm since the pruning of candidate keys is closer to the first k+1 while implementing the modified one takes a lot of execution time since the pruning starts with the k(n) – 1 where n is the maximum set size with set size frequency >= minimum support.

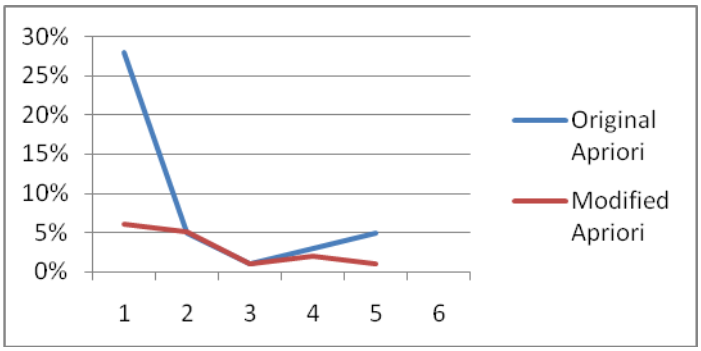

Fig. 1. Execution Time of Original vs. Modified Apriori

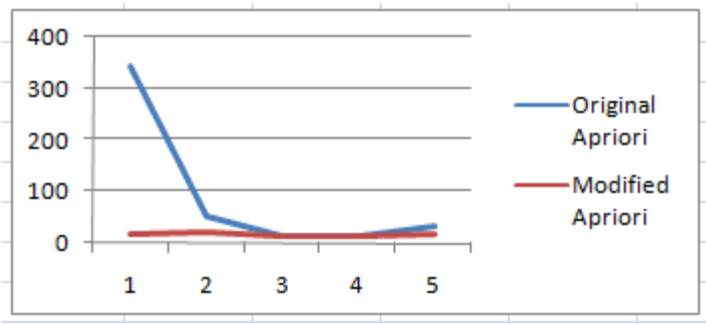

Fig.2. No. of DB Passes of Original vs. Modified Apriori

The graph shows that in terms of execution time, the original apriori executes more time compared to the modified one. Moreover, in terms of database passes, the modified apriori provides less database access compared with the original one that makes its execution faster.

7 Conclusion

There are several ways to improve the database access of Apriori algorithm thereby improving also the efficiency of the execution. Based on the modified code, set size and set size frequency were introduced. These factors helped in a more rapid generation of possible association of frequent items. In terms of database passes, the modified apriori provides less database access compared with the original one that makes its execution faster.

8 Future Work

Suggestions in finding other ways to generate combinations are encourage. Currently, further research in finding a faster way of pruning candidate keys is undergoing in finding the ideal starting size of combination size.

References

[1] D. Kerana Hanirex, and M.A. Dorai Rangaswamy. 2011. Efficient Algorithm for Mining Frequent Itemsets using Clustering Techniques. *International Journal on Computer Science and Engineering (IJCSE)* Vol. 3 No. 3 March 2011.

[2] E. Ramaraj and N. Vankatesan, " Bit Stream Mask Search Algorithm in Frequent Itemset Mining," European Journal of Scientific Reasearch," Vol. 27 No. 2 (2009), pp. 286-297

[3] J. Han, M. Kamber, "Data Mining: Concepts and Techniques," Morgan Kauffman, San Francisco, 2000

[4] J.5 Han, J. Pei and Y.Yin, "Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach," In Proceedings ACM-SIGMOD International Conference Management of Data (SIG MOD'04), pages 53-87, 2004

[5] J. Hossen, A. Rahman, K. Samsudin, F. Rokhani, S. Sayeed, and R. Hasan, " A Novel Modified Adaptive Fuzzy Inference Engine and Its Application to Pattern Classification, World Academy of Science, Engineering and Technology 80, 2011

[6] M. H. Marghny and A.A. Mitwaly, "Fast Algorithm for Mining Association Rules," In proceedings of the First ICGST International Conference on Artificial Intelligence and Machine ELearning AIML05, pages 36-40, Dec. 2005

[7] Mamta Dhanda, "An Approach to Extract Efficient Frequent Patterns from transactional database," International Journal of Engineering Science and Technology (IJEST), Vol.3 No.7, July 2011, pp. 5652-5658

[8] P.Purdon, D. Gucht and D. Groth," Average Case Performance of the Apriori Algorithm," Society for Industrial and Applied Mathematics (2004) Vol. 33 No.5 pp. 1223-1260

[9] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in Proceedings of the 20th VLDB Conference, 1994, pp. 487-499

[10] R. Agrawal, T. Imielinski and A. Swami, "Mining association rules between sets of items in large databases." SIGMOD'93, 207-216, Washington, D.C.

[11] S. Murali Krishna and S. Durgs Bhavani, " An Efficient Approach for Text Clustering Basaed on Frequent Itemsets," European Journal of Scientific Research, Vol. 42 No. 3 (2010), pp.385-396

[12] S. Prakash and R.M.S. Parvathi, "An Enhanced Scaling Apriori for Association Rule Mining Efficiency," European Journal of Scientific Research, Vol. 39 No. 2 (2010), pages 257-264