International Journal of Aquatic Science
ISSN: 2008-8019
Vol 12, Issue 02, 2021

DISCOVERING RARE ITEMSETS FROM TRANSACTIONAL DATABASE USING BINARY ENHANCED APRIORI – BEAP ALGORITHM

R.Savitha[1], Dr.V.Baby deepa[2],
[1]*Research Scholar, Government Arts College (autonomous), Karur,*
[2]*Research Advisor, Government Arts College (autonomous), Karur*
savithbca@gmail.com[1], deepamct@gmail.com[2]

ABSTRACT
An itemset is an occurrence pattern that is hidden underneath in the raw data and produced after mining the database to aid the user to make some decision to improve their business needs. Most of the previous explorations are carried out in extracting the frequently occurring itemsets or patterns and the rare patterns are often neglected. The rare patterns are very useful in many areas like fraud detection, less selling products, intrusion attack on networks and this rare patterns mined will signify the user with some interesting details to overcome and evade those issues. This paper proposes a new algorithm named BEAP algorithm that utilizes the binary representation and discover the rare itemset without compromising on the speed and reduces the memory consumption. From the experimental evaluation the proposed algorithm BEAP is proved to be effective and efficient than the existing algorithms.

1. INTRODUCTION

Due to surge and the evolution of the internet, the data available to the business firms and the individual is on the rise. To extract useful data patterns (knowledge) from the junk data is called data mining and these extracted patterns are provided to some other process to decode and interpret them to the common man. The patterns mined will be in textual format or graph format and extracted from plethora of databases like sequential, transaction and timestamp databases. Depending upon the pattern, the type of mining is employed and utilized.

The usual pattern mining approach is designed to unearth the most frequent occurring items in the data and this FI usually reveals the symmetries present in the data. To be more concise theyendorse the often repetitive occurring items in the database whereas the rare itemset does not occur often but they are interesting in many areas like medical diagnosis and computer network attacks. Hence the rare ones are often called abnormal or anomaly in user purchasing behavior [1] and they denote the suspicious in security of the system[2]. Since the rare pattern mining is quite cumbersome to extract the results, the mining algorithms need some distinctive care and customized tailor made solutions. The rare itemset mining is a new area and the problems involved in the operational process is not studied and analyzed yet.

The data mining is a process of fetching the hitherto not known, obscured, valuable, and beneficial patterns from a huge amount of unstructured raw data available across the world.

2. BASIC CONCEPTS

Let us consider a set of k items (i.e.) $I = \{a_1,a_2,a_3, \ldots a_k\}$ and TID = $\{T_1,T_2,T_3.. T_K\}$. A subset of Y of I is denoted as an itemset if $|Y| = k$, then Y is a k-itemset. For example a_1a_2 is a two itemset of I. The items $\{a_1, a_2, a_3\}$ are identified by unique identifier $\{T_1, T_2, T_3\}$.. the support is the actual measure or the count where the items are present in the transaction(frequency). If a given itemset $\{a_1,a_2\}$ count is greater than or equal to the support count value provided by the user, then the itemset is considered as frequent or else it is considered as infrequent or rare. The sample dataset is given in the table 1. The lattice set of the sample database is shown in the figure 1.

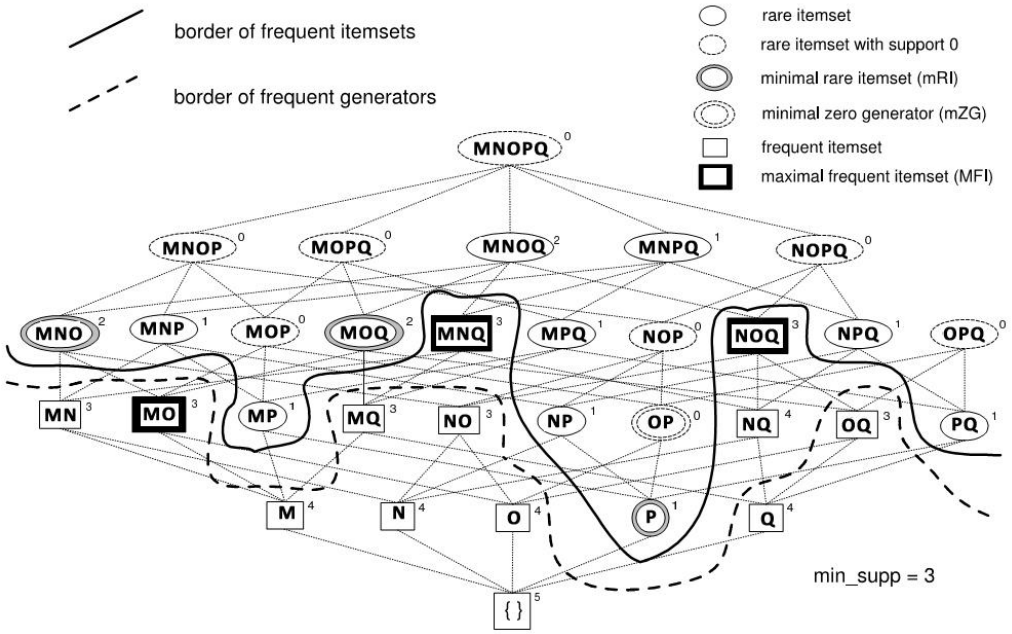

Figure 1: lattice of the sample database shown in the table 1

TID	ITEMS
1	M, N, P, Q
2	M, O
3	M, N, O, Q
4	N, O, Q
5	M, N, O, Q

Table 1: Sample database

3. OBJECTIVE OF THIS PAPER

The main objective of this research paper is to develop a new algorithm named "Binary Enhanced APriori – BEAP algorithm" by combining the binary representation and discover the rare itemsets from large databases. The secondary objectives are illustrated below,

1. To make a detailed comparison with the existing algorithms with respect to run time and memory consumption using some benchmarked databases.
2. To analyze and scrutinize the existing algorithms which deals with the mining of rare itemsets using Apriori approach.

4. PROBLEM BACKGROUND

Even though the mining of frequent itemset is evolved to a great extent, the discovery of the frequent and rare itemsets from the database is still a cumbersome task as it deals with lot of computations and operations related support count and pruning of the unpromising items. The foremost problem is the generation of the large numbers of frequent itemset and this makes the algorithm inefficient. If the minimum support value is set to a very low value, the volume of candidates generated will be high and if the minimum support value is set to a high value, the rare items will be missed. This problem is fixed in the proposed work by introducing binary representation and starting to find the rare items from the unique item combination and there by the proposed algorithm will deal with only 2^U combinations that is U is the unique items.

5. FIND THE UNIQUE ITEMS FROM THE DB

Initially the unique items from the sample database shown in the table 1 are found using the procedure FindUni shown in the figure 2.

PROCEDURE FindUni (Database D)

ITEM	T1	T2	T3	T4	T5
M	1	1	1	0	1

INPUT: Database D
OUTPUT:Unique Items
BEGIN:

1. Load the input database D into the procedure FindUni
2. Initialize an array G[ ] = ϕ
3. Find the total number of rows in the DB ➔T
4. *For each* row $r_i \in$ R
5. *For each* Item $I_i \in r_i$
6. *if* [I_i Not IN G[ ]] *do begin*
7. G[ ] ← I_i*// store unique items*
8. Else
9. Count I_i value ➔ I_i counter *// count unique items*
10. Close IF
11. Close For
12. Close For

Return G[ ] with count
END

Figure 2: Pseudo code of the procedure to find the unique items

The figure 2 showcased the procedure to find the unique items from the database and the output of the procedure is shown in the following table 2.

ITEMS	FREQUENCY
M	4
N	4
O	4
P	1
Q	4

Table 2: Unique items discovered

6. REPRESENT THE DATA IN BINARY

The database is reconstructed in binary format that is if an item is present in the row identifier, then that item is marked with "1" and if it is not present in the row identifier it is marked with "0". For example, the unique item {M} is present in the TID's {T1, T2, T3, T5} and marked as,

The procedure to reconstruct the database into binary format is shown in the following figure 3.

PROCEDURE ReconstructDB(database D, Unique G)

INPUT: Database D, Unique items G
OUTPUT: Reconstructed DB in binary format
BEGIN:

1. Load the input database D into the procedure
2. Load the Unique G into the procedure
3. Initialize Binary[ ] = ϕ
4. *For each* row $r_i \in$ R
5. *For each* Item $I_i \in$ G
6. *if* [I_i= r_i] *do begin*
7. Mark I_i➔ "1" and store in Binary
8. Else
9. Mark I_i➔"0" and store in Binary
10. Close IF
11. Close For
12. Close For

```
Return Binary[ ]
END
```

Figure 3: Pseudo code of the procedure to reconstruct the DB in binary format

The figure 3 showcases the procedure to reconstruct the database into binary format using the following property,

"If an item Xi is available in the row identifier i, that item Xi = 1 else Xi = 0"

The following table 3 shows the reconstructed database and this database is comprised of only ones and zeroes.

ITEM	T1	T2	T3	T4	T5	COUNT
M	1	1	1	0	1	**4**
N	1	0	1	1	1	**4**
O	0	1	1	1	1	**4**
P	1	0	0	0	0	**1**
Q	1	0	1	1	1	**4**

M	1	1	1	0	1
N	1	0	1	1	1
O	0	1	1	1	1
P	1	0	0	0	0
MNOP	0	0	0	0	0

Table 3: Reconstructed database

7. PERMUTE FROM THE TOP LEVEL ITEMSET

The permutations starts from the top level (i.e.) {M, N, O, P, Q} and the frequency or the count of this itemset is calculated from the reconstructed table 3 using simple AND operation on this binary values as shown in the following section,

The itemset {{M, N, O, P, Q} is a rare itemset with support 0. Now the next level itemsets are permuted and the calculations are shown in the following section,

M	1	1	1	0	1
N	1	0	1	1	1
O	0	1	1	1	1
Q	1	0	1	1	1
MNOQ	0	0	1	0	1

The 4-itemset {M, N, O, P}, {M, N, O, Q}, {M, N, P, Q}, {M, O, P, Q}, {N, O, P, Q} and the count for these items are calculated using the binary AND.

M	1	1	1	0	1	
N	1	0	1	1	1	
O	0	1	1	1	1	
P	1	0	0	0	0	
Q	1	0	1	1	1	**AND**
MNOPQ	0	0	0	0	0	

The itemset {M, N, O, P} is a rare itemset with support 0. The next item is computed.

The itemset {M, N, O, Q} is a rare itemset with support =2. This is a non-zero rare itemset. The next itemset is computed.

The itemset {M, N, P, Q} is a rare itemset with support =1 and this is a non-zero rare itemset.

M	1	1	1	0	1
N	1	0	1	1	1
P	1	0	0	0	0
Q	1	0	1	1	1
MNPQ	1	0	0	0	0

M	1	1	1	0	1
O	0	1	1	1	1
P	1	0	0	0	0
Q	1	0	1	1	1
MOPQ	0	0	0	0	0

M	1	1	1	0	1
N	1	0	1	1	1
O	0	1	1	1	1
MNO	0	0	1	0	1

M	1	1	1	0	1
N	1	0	1	1	1
P	1	0	0	0	0
MNO	1	0	0	0	0

The itemset {M, O, P, Q} is a rare itemset with support =0 and this is a zero rare itemset.

The itemset {N, O, P, Q} is a rare itemset with support =0 and this is a zero rare itemset. Now the next level 3-itemset permutation is computed as shown in this section.

The 3-itemsets are {M, N, O}, {M, N,P}, {M, N, Q}, {M,O, P}, {M, O, Q}, {M,P,Q}, {N,O,P},{N,O,Q}, {N, P,Q},{O,P,Q}.

First {M, N, O} support is computed as shown in below,

N	1	0	1	1	1
O	0	1	1	1	1
P	1	0	0	0	0
Q	1	0	1	1	1
NOPQ	0	0	0	0	0

The itemset {M, N, O} is a rare itemset with support =2. This is a non-zero rare itemset. The next itemset is computed.
The itemset {M, N, P} is a rare itemset with support =1. This is a non-zero rare itemset. Similarly all the permutations are fetched and computed. The final results is shown in the table 4.

ITEMS	COUNT	ITEMS	COUNT
MNOPQ	0	MPQ	1
MNOP	0	NOP	0
MNOQ	2	NPQ	1
MNPQ	1	OPQ	0
MOPQ	0	MP	1
NOPQ	0	NP	1
MNO	2	OP	0
MNP	1	PQ	1
MOP	0	P	1
MOQ	2		

Table 4: Final output of rare itemset

8. EXPERIMENTAL RESULTS

The entire experiment is executed on dual core processor with 4GB RAM at 2GHz and the datasets are collected from the following domain http://fimi.cs.helsinki.fi/data/.

The proposed algorithm is developed using java SMPF tool and compared with other existing algorithms like Apriori-Inverse [4] and ARIMA[5] and the results related to the run time and memory consumption is shown in the following graphs. The synthetic database is generated by the IBM-Quest tool and the algorithms are executed on the synthetic database.

The characteristics of the benchmarked database are shown in the following table 5 and these databases are used for execution of the algorithms.

Dataset	Size (MB)	# Items	# Trans	AvgTrans
Mushroom	19.3	119	8124	23
Retail	4.2	16,470	88,126	10.3
Pumsb	16.3	2,113	49046	74
Kosarak	30.5	41,271	990,002	8.1

Table 5: The characteristic of the database

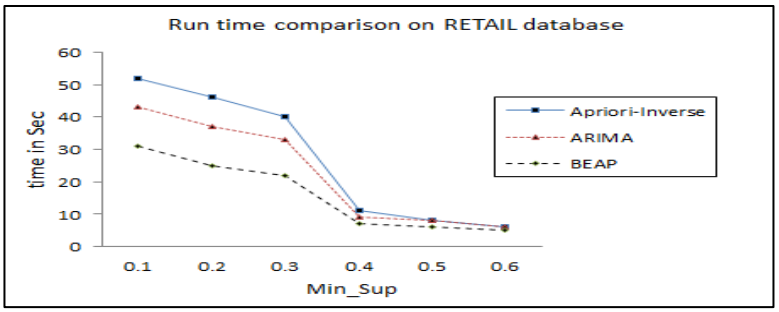

Figure 4: Comparison graph for run time on retail database

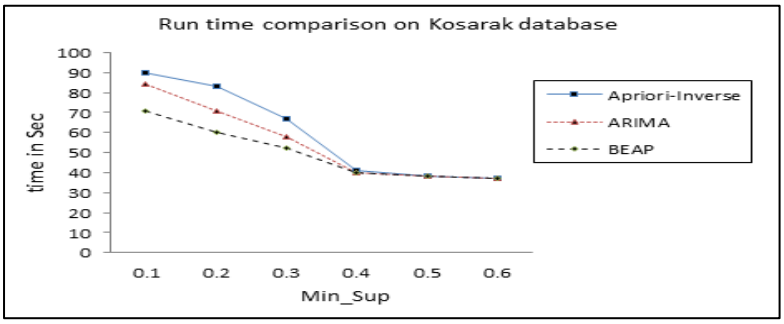

Figure 5: Comparison graph for run time on retail database

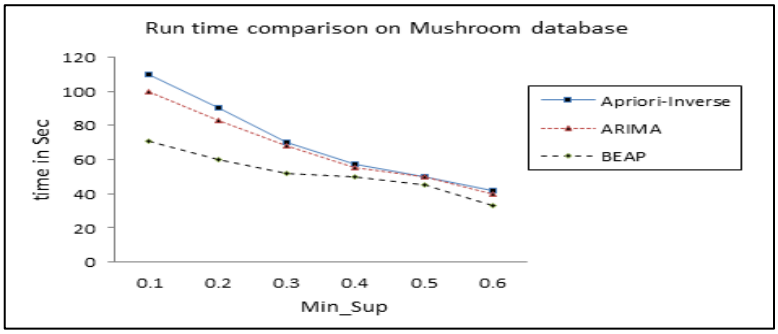

Figure 6: Comparison graph for run time on mushroom database

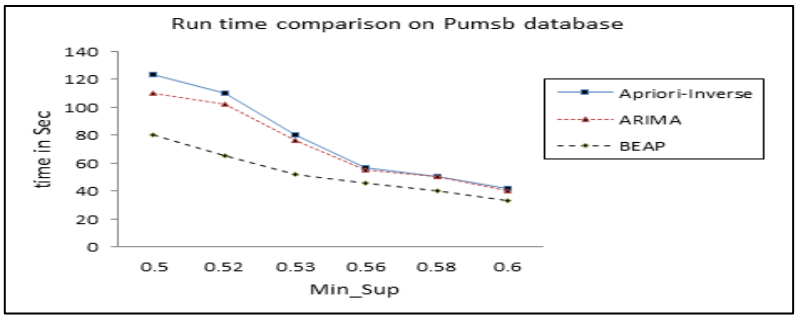

Figure 7: Comparison graph for run time on pumsb database

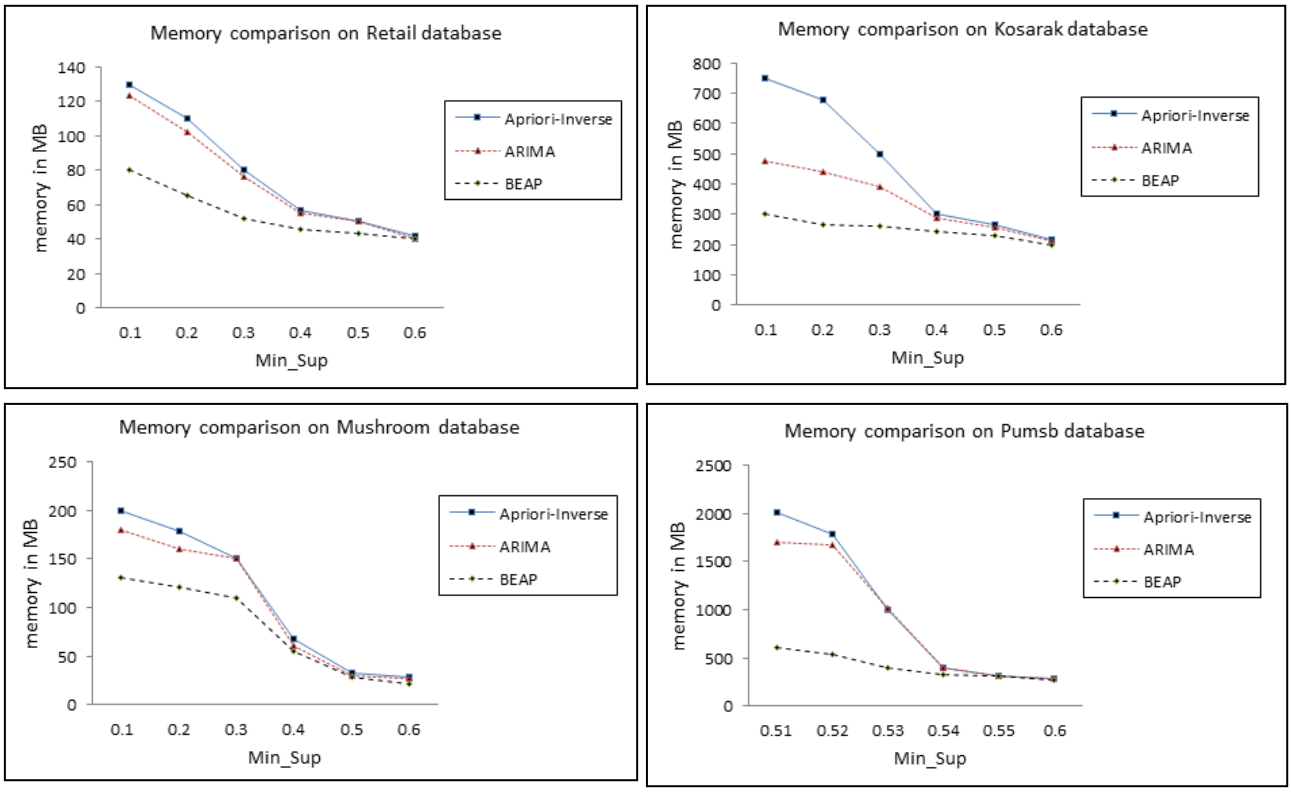

Figure 8: Memory consumption comparison graphs

From the figures 4, 5, 6, 7 it is quite evident that the proposed BEAP algorithm outscored the other existing algorithms by a good margin with respect to run time. The main reason for the proposed algorithm performance is it considered only the rare itemsets and ignored the other itemsets which saved the execution time considerably and outscotred the other two algorithms. The next section shows the memory consumption.

From the figure 8, it is obvious that on sparse database like kosarak the Aprioiri-inverse performed very badly and one the min_sup is above 0.5, all the algorithms performed equally well. When the algorithms are executed on the dense databases, the proposed BEAP algorithm outperformed the other two algorithms bya huge margin. Notably the proposed BEAP algorithm consumed very less memory for all minimum support threshold values as shown in the figure 8.

9. CONCLUSION

In this paper a new algorithm named BEAP is proposed to discover rare itemset and the proposed BEAP algorithm employed binary representation and bottom up approach to find the rare itemset with very less execution time and memory usage. The experimental results showcased that the

proposed BEAP algorithm outscored the other two algorithms by a good magnitude regarding the speed and memory footprints.

REFERENCES

[1] E. Suzuki. Undirected Discovery of Interesting Exception Rules. International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI), 16(8):1065–1086, 2002.
[2] Wenke Lee and Salvatore J. Stolfo, "Data Mining Approaches for Intrusion Detection", *Proceedings of the 7th USENIX Security Symposium*, *San Antonio*, *Texas*, 1998.
[3] G. Weiss. Mining with rarity: a unifying framework. SIGKDD Explor. Newsl., 6(1):7–19, 2004.
[4] Y. Koh and N. Rountree. Finding Sporadic Rules Using Apriori-Inverse. In Proc. of PAKDD '05, Hanoi, Vietnam, volume 3518 of LNCS, pages 97–106. Springer, May 2005.
[5] Laszlo Szathmary, Amedeo Napoli, and PetkoValtchev, "Towards Rare Itemset Mining," *19th IEEE International Conference on Tools with Artificial Intelligence*, *Patras*, *Greece. 1*, 2007, pp. 305-312.