Proposed Improved FP-Growth Algorithm with Multiple Minimum Support Threshold Value (MISIFP-Growth) For Mining Frequent Itemset

M.Sinthuja
Research Scholar
Annamalai University
Chidambaram, India
sinthujamuthu@gmail.com

Dr. N. Puviarasan
Associate Professor
Annamalai University
Chidambaram, India
npuvi2410@yahoo.in

Dr. P.Aruna
Professor and Head
Annamalai University
Chidambaram, India
arunapuvi@yahoo.co.in

Abstract- Frequent pattern mining is a key step in many association rule mining algorithms. In the basic model of association rules, a pattern is said to be frequent if it satisfies the user-defined minimum support (minsup) threshold value. Since only a single minsup is used in the entire database, the basic model of frequent patterns leads to the problem known as “rare item problem” which is as follows: at high minsup, we miss the frequent patterns containing rare items, and at low minsup, combinatorial explosion can occur, producing too many frequent patterns. To confront the rare item problem, an effort has been made in the literature to find frequent patterns with “multiple minimum supports framework.” In this framework, each item is given a constraint known as minimum item support (MIS). The notion of minimum support for a pattern is defined as the minimal MIS value among all its items. Efforts are being made to propose “IFP-Growth” based approach to extract patterns under “multiple minsup framework”. This generalized framework enables the user to simultaneously specify high minsup for a pattern containing only frequent items and low minsup for a pattern containing rare items. Experimental results reveal that proposed MISIFP-Growth algorithm outperform FP-Growth algorithm in terms of execution time, memory usage.

Index Terms- Data Mining, frequent itemset, IFP-Growth, Minimum support, Multiple Minimum Support

1. INTRODUCTION

The vital logic that people were attracted by IT is the discovery of useful information from huge collection of data industry towards the domain of “Data Mining” [1] [2] [3] [4]. From the huge data, we barely explore useful knowledge for decision analysis in the business. Vast collection of data can be in distinct formats like audio, video, numbers, text, figures, and hypertext formats. To perform data mining task expertise and learning are fundamental need because the victory and loss of data mining projects is highly dependent on the person who are administrating the procedure due to lack of standard protocol. The lifecycle of data mining is of six steps they are Data cleaning, Data integration, Data Selection, Data transformation, Data Mining, Knowledge discovery.

Frequent pattern mining, which is the most important field in association rule mining, was first introduced for Market Basket Analysis [4] [8] [9][10]. The goal of frequent pattern mining is to discover frequent patterns whose support is greater than or equal to the minimum support threshold. Pattern mining algorithm can be enforced on various data such as transaction databases etc. Frequent Patterns are itemsets, substructures that appear in a database with high frequency. They are Candidate generation and Pattern growth.

2. RELATED WORKS

In the section, three algorithms, including the Apriori algorithm, the MSapriori algorithm and the FP-growth algorithms, are briefly reviewed. The Apriori algorithm is the most popular algorithm for mining frequent itemsets. However, it has two problems: (1) it only allows a single MS threshold, and (2) its efficiency is usually not satisfactory. As to the first problem, the MSapriori algorithm extends the Apriori algorithm so that it can find frequent patterns with multiple MS thresholds. As for the second problem, many algorithms have been proposed to improve the efficiency. The FP-growth algorithm is one of these improved algorithms and is probably the most well-known. The FP-growth algorithm contains two phases, where the first phase constructs an FP-tree, and the second phase recursively projects the FP-tree and outputs all frequent patterns.

2.1. ***Frequent Pattern with Single Threshold:*** Let D be a transaction database over a set of items I, and *minsup* is minimum support threshold given by the user. The set of frequent patterns in D are the patterns which exceed *minsup.* As an example, suppose there are two itemsets: K = {x, y, z} and Z = {n, m} with actual support = 70%, 40%, respectively in a given database and the *minsup* is set at 50%. According to given definition above, the K itemset is frequent as its support exceeds *minsup* = 50%, whereas Z is infrequent its support does not satisfy *minsup*.

2.2. ***Frequent Pattern with Multiple Thresholds:*** Let I be a set of I items I = {i_1,…, i_n}, an itemset X = {i_1, …, i_k}, the minimum item support (MIS) of itemset X is defined as follows: MIS(X)=MIN{MIS(I_1),MIS(I_2),…,MIS(I_k)}. As an example assume that an itemset K = {x, y, z} has an actual support = 8% in a given database. Suppose that the MIS of items are given as: MIS(x) = 5%, MIS(y) = 10%, MIS(z) = 15% and actual supports of items are given as: sup(x) = 10%, sup(y) = 9%, sup(z) = 11% then the MIS of the itemset K can be defined as: MIS(KK)=MIN {MIS(xx)=5%, MIS(yy)= 10%, MIS(zz)=15%} = 5%. Thus, the itemset K is frequent with support = 8%, which exceeds MIS of K = 5%. This is called downward closure property with MIS [5] [6] [7] [11] [12] [13]; in another words, any itemset containing an item with support less than the lowest minimum support threshold cannot be considered as frequent.

3. PROPOSED FREQUENT ITEMSET MINING FOR MULTIPLE MINIMUM SUPPORTS BASED ON IMPROVED FP-GROWTH ALGORITHM

In this section, the proposed algorithm of MISIFP-Growth is introduced to mine frequent itemsets with multiple support thresholds. The bottom-up tree based algorithm, MISIFP-growth, is revisited to be self-contained. We start by explaining a motivating example that will be used in the presentation of the algorithms. A sample database is given in the Table 1. Multiple support threshold of each item is given in the Table 2. Last row of Table 2 shows actual support of each item in the database D. In the right most column of Table 1, items in the transactions are in decreasing order of their multiple support thresholds.

In MISIFP-Growth algorithm, bottom up approach is utilized to mine frequent itemsets. Bottom-up strategy builds itemset combinations from smallest to the largest like FP-growth. Major difference between FP-Growth and the proposed MISIFP-Growth algorithm is the capability of finding frequent itemsets based on multiple support thresholds rather than particular minimum support given by the user. MISIFP-Growth is of two steps

1. Construction of pattern growth tree
2. Generating frequent patterns from the tree

Discarding property is used in this algorithm. Any item that has support lower than minimum of MIS (MIN-MIS) is discarded and is not used. Let us go through our motivating example to explain this property. The least minimum support threshold in this example is 2 as seen from MIS values in the second column of Table 2. By utilizing pattern growth tree called MISIFP-Growth (Multiple Item Support Frequent Pattern growth) frequent itemset can be explored by considering multiple support threshold value. This tree is established by scanning all the transactions that exist in the transaction database.
The steps of MISIFP-Growth algorithm can be understood by the example.

Scan the database D once to find out the support of each item as shown in the second row of Table 2. Find out the least minimum support threshold among all minimum item support thresholds: MIN-MIS=2. Once again scan the database to build MISIFP-tree with the items present in the right column of the Table 1. The process of insertion is as follows.

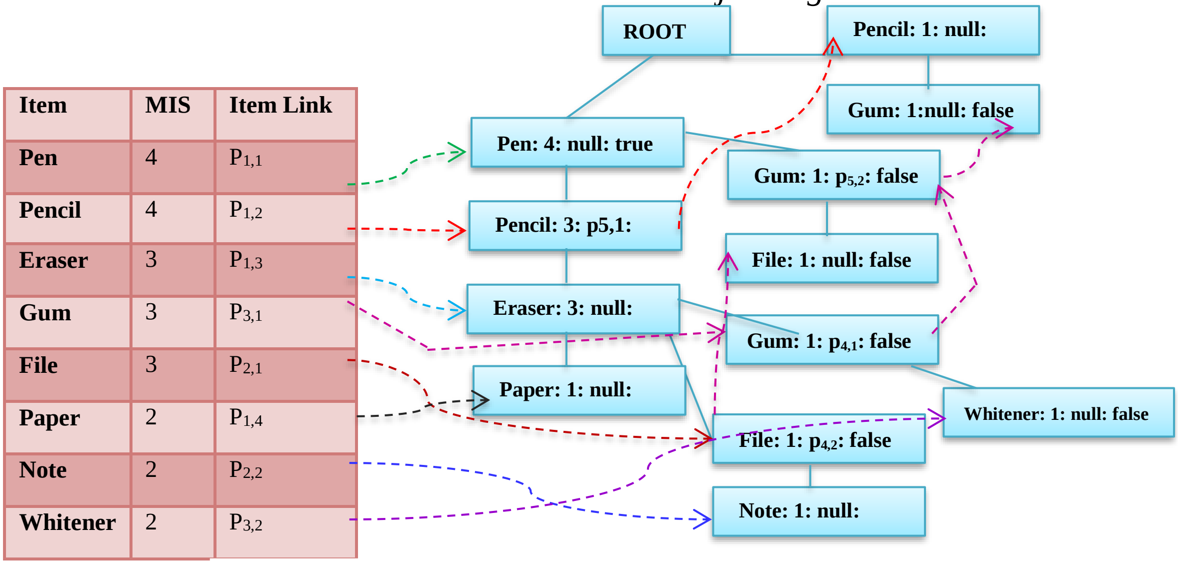
The root of MISIFP-tree is generated and specified as "null". Each transaction is inserted into the tree in terms of descending order that have support greater than or equal to 2.

Table 1. Transaction Database

TID	Item bought	Ordered item
100	Paper, Pencil, Eraser, Pen	Pen, Pencil, Eraser, Paper
200	File, Pencil, Eraser, Pen, Note	Pen, Pencil, Eraser , File, Note
300	Gum, Eraser, Pencil, Pen, Whitener	Pen, Pencil, Eraser, Gum, Whitener
400	File, Gum, Pen	Pen, Gum , File
500	Pencil, Gum	Pencil, Gum

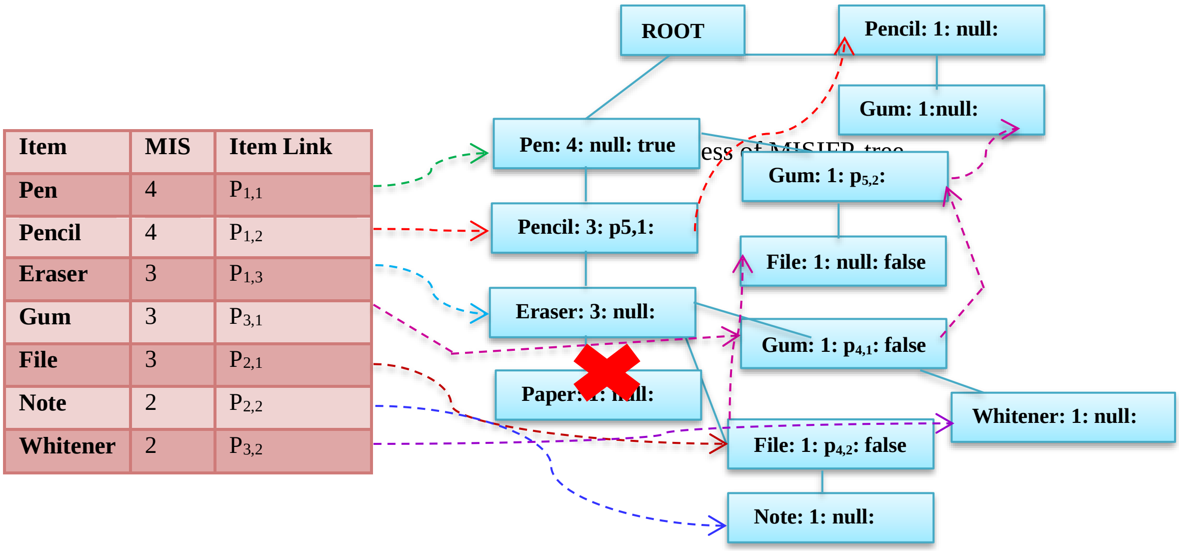
Table 2. Prioritize the items

Items	MIS	Actual Support
{Pen}	4	4
{Pencil}	4	4
{Eraser}	4	3
{Gum}	3	3
{File}	3	2
{Paper}	2	1
{Note}	2	1
{Whitener}	2	1



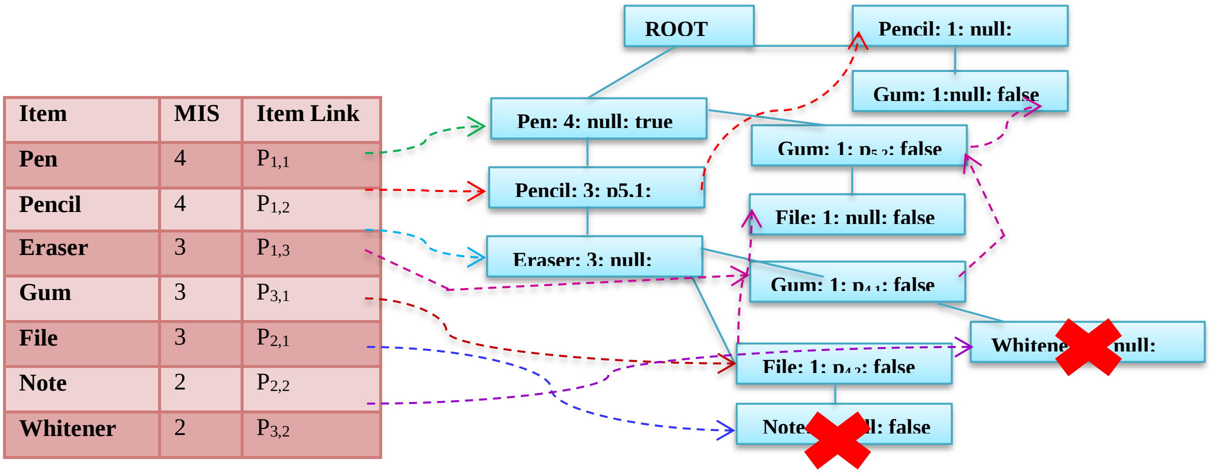
Item	MIS	Item Link
Pen	4	$P_{1,1}$
Pencil	4	$P_{1,2}$
Eraser	3	$P_{1,3}$
Gum	3	$P_{3,1}$
File	3	$P_{2,1}$
Paper	2	$P_{1,4}$
Note	2	$P_{2,2}$
Whitener	2	$P_{3,2}$

Figure 1. Incomplete MISIFP-tree



Item	MIS	Item Link
Pen	4	$P_{1,1}$
Pencil	4	$P_{1,2}$
Eraser	3	$P_{1,3}$
Gum	3	$P_{3,1}$
File	3	$P_{2,1}$
Note	2	$P_{2,2}$
Whitener	2	$P_{3,2}$

Figure 2. Pruning item Paper from MISIFP-tree



Item	MIS	Item Link
Pen	4	$P_{1,1}$
Pencil	4	$P_{1,2}$
Eraser	3	$P_{1,3}$
Gum	3	$P_{3,1}$
File	3	$P_{2,1}$
Note	2	$P_{2,2}$
Whitener	2	$P_{3,2}$

Figure 3. Pruning item Note and Whitener from MISIFP-

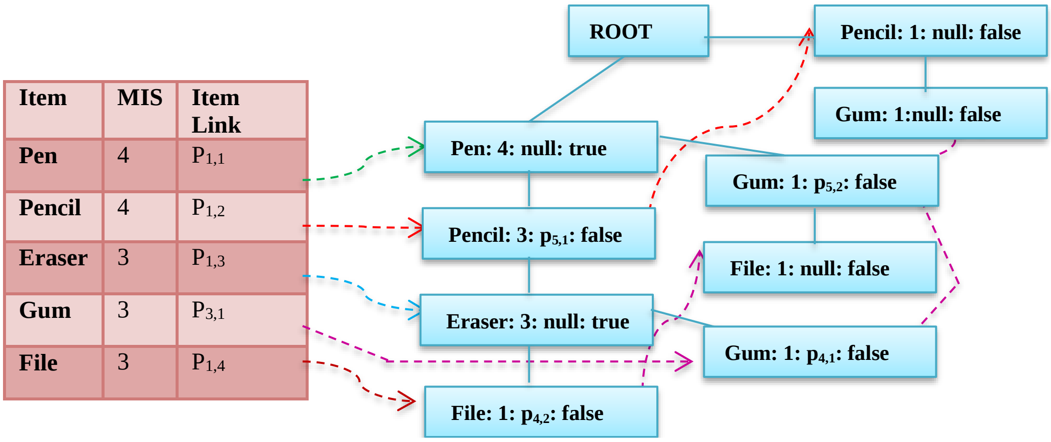


Figure 4. Complete proposed MISIFP-tree

First transaction of itemset {Pen, Pencil, Eraser and Paper} is inserted into MISIFP-tree. The count for each node is assigned by 1. Nodes that have the same item-name are linked in order by the pointer of node-links starting from head of node-link of header table as seen in Fig.1. Flag value is fixed to false as it don't have any branch node. Node link in the header table is also to be updated. For the second transaction {Pen, Pencil, Eraser, File and Note}; since it shares the prefix {Pen, Pencil and Eraser} with the first transactions, the count of each node along the prefix is increased by 1, a new node {File and Note: 1} is generated and linked as child of {Eraser: 2}. Simultaneously node link and flag is also updated. By repeating same steps, ensuing transactions are added to the tree. Discarding property is used in this algorithm. Any item that has support lower than minimum MIS (multiple minimum support) is discarded and that item in tree and is not used anymore. With reference to Fig 2 item "Paper" is eliminated whose support is lower than minimum MIS. Likewise item " Note and Whitener" is also deleted as its support is lower than minimum MIS as shown in Fig 3 and 4. Complete proposed MISIFP-tree is shown in Fig. 5. Complete set of frequent itemset is mined using multiple minimum support threshold value as shown in Table 3.

Table 3. Generation of Frequent itemsets

Items	MIS	Conditional pattern base	Conditional MISIFP-tree	Frequent patterns generated
{Pen}	4	-	-	No
{Pencil}	4	{Pen: 3}	-	No
{Eraser}	3	{Pen :3 Pencil: 3 }	{ Pen, Pencil: 3}	{ Pen, Pencil, Eraser:3}

{Gum}	3	{Pen:1 Pencil: 1 Eraser: 1}
		{ Pen:1 }{ Pencil: 1 }
			-	No

{File}	3	{Pen:1}{Gum:1}
		{Pen:1 Pencil: 1 Eraser: 1}
			-	No

{Paper}	2	-	-	
{Note}	2	-	-	{f, c:3}
{ Gum }	2	-	-	No

4. PERFORMANCE EVALUATION

In this section, the proposed algorithm of MISIFP-Growth is compared with the existing tree based algorithm of FP-Growth to discover frequent itemset under multiple minimum support thresholds. Numerous experiments are conducted using two dataset namely Mushroom and Kosarak to verify the effectiveness and efficiency of the proposed algorithm. In these experiments, the performance of consumption of time and memory are measured.

4.1 Experimental environment and datasets

Experiments are conducted using two datasets to calculate the performance of the proposed MISIFP-Growth. The experiment is conducted on Intel® corei3™ CPU, 2.13 GHz, and 2GB of RAM computer. Implementation is done in Java. Characteristics of datasets is shown in Table 4.

Table 4. Characteristics of datasets

Datasets	**Density (%)**	**Size (MB)**	**# of distinct items**	**Average Transaction Length**	**# of Transactions**
Mushroom	19.3	0.56	119	23	8124
Kosarak	0.002	0.5	41271	8.1	990002

4.2 Execution Time

In this subsection, tree structure algorithms of proposed MISIFP-Growth are compared with FP-Growth algorithm. From the graph y-axis display runtime in millisecond and x-axis display the different minimum support values For the dataset Kosarak both algorithms performs good but the proposed MISIFP-Growth algorithm is about 250 orders of magnitude faster than FP-Growth algorithm as shown in Fig. 5.

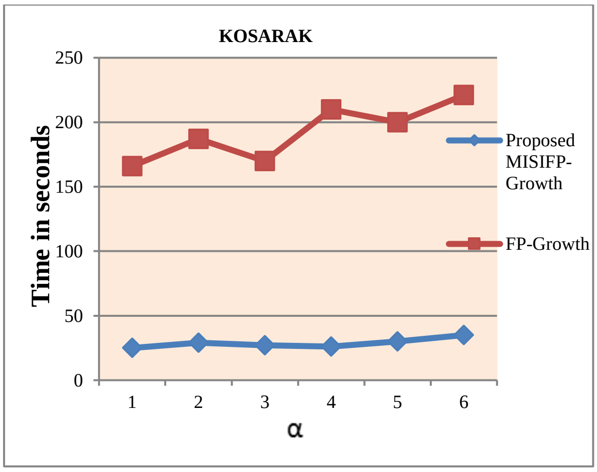


Fig. 5. Runtime of proposed MISIFP-Growth algorithm

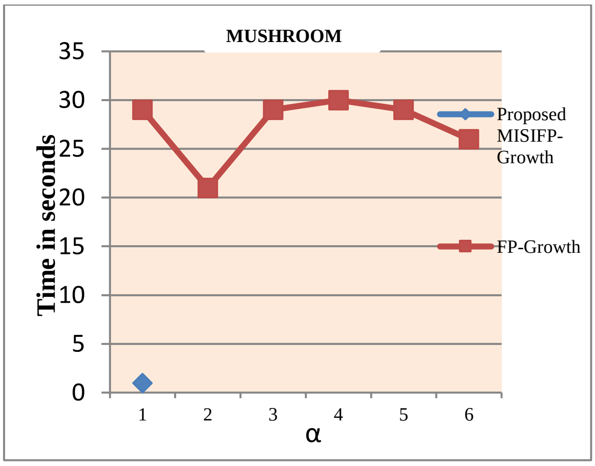


Fig. 6. Runtime of proposed MISIFP-Growth algorithm

The proposed MISIFP-Growth performs better than FP-Growth algorithm for all α values. For the dataset Mushroom MISIFP-Growth algorithm is 25 times faster than FP-Growth algorithm as shown in Fig. 6. This shows that the performance of the proposed MISIFP-Growth algorithm is outstanding when compared to FP-Growth algorithm.

4.3 Consumption of Memory

In this subsection, the proposed MISIFP-Growth algorithm is compared with FP-Growth algorithm. From the graph longitudinal axis shows the memory in MB and latitudinal axis shows the different support threshold values.

For the dataset Kosarak memory consumption of FP-Growth algorithm is highest one when compared to the proposed MISIFP-Growth algorithm. Consumption of memory of FP-Growth algorithm is three to four times higher the proposed MISIFP-Growth algorithm as portrayed in Fig 7. The comparison of performance of the database Mushroom is portrayed in Fig. 8.

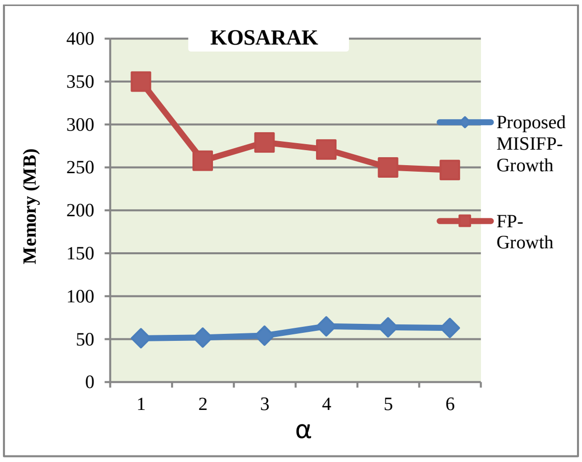


Fig. 7. Memory usage of proposed MISIFP-growth algorithm

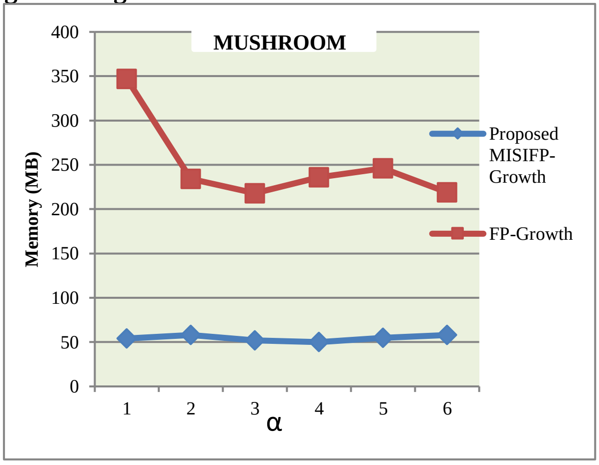


Fig. 8. Memory usage of proposed MISIFP-growth algorithm

It is observed from the graph that the consumption of memory of MISIFP-Growth algorithm is lower when compared to FP-Growth algorithm. This shows that the performance of the proposed MISIFP-Growth algorithm is better than FP-Growth algorithm.

5. CONCLUSION

To mine frequent itemset which contains both frequent and rare items we use "Multiple Item Support". By utilizing this Multiple Item Support MISIFP-Growth algorithm have been proposed to mine frequent itemsets. Proposed MISIFP-Growth algorithm is based on FP-Growth algorithm. All the necessary information are holed in MISIFP-tree which are useful in the process of mining. This tree contains useful information that plays an important role in mining frequent and rare itemsets. This paper analyses the behavior of proposed MISIFP-growth and FP-Growth algorithms with datasets of different characteristics. From the experimental results, it is showed that the proposed MISIFP-Growth produced an outstanding performance in terms of runtime and memory usage.

Acknowledgement. The authors would like to thank the reviewers for their valuable suggestions which help in improving the quality of this paper. We would like to thank the Department of Computer Science and Engineering, Annamalai University for providing the lab facilities. We gratefully acknowledge the funding agency, the University Grant Commission (UGC) of the Government of India, for providing financial support for doing this research work.

REFERENCES

[1] J. Han, J. Pei, Y. Yin. 2000. Mining frequent patterns without candidate generation, Proceedings ACM-SIGMOD International Conference on Management of Data (SIGMOD' 00), Dallas.

[2] R. Agrawal and R. Srikant. 1994. Fast algorithms for mining association rules in large databases. In Proceedings of the 20th International Conference on Very Large Data Bases, VLDB '94, pages 487–499.

[4] J. Han, H. Cheng, D. Xin, and X. Yan.2007. Frequent pattern mining: current status and future directions. Data Min. Knowl. Discovery, 15(1):55–86.

[4] J. Han, J. Pei, Y. Yin, and R. Mao. 2004. Mining frequent patterns without candidate generation: A frequent- pattern tree Approach. Data Min. Knowl. Discov. 8(1):53–87.

[5] Y.H. Hu and Y.-L. Chen. 2006. Mining association rules with multiple minimum supports: a new mining algorithm and a support tuning mechanism. Decis. Support Syst., 42(1):1–24.

[6] B. Liu, W. Hsu, and Y. Ma. 1999. Mining association rules with multiple minimum supports. In KDD '99: Proceedings of the Fifth ACM SIGKDD international conference on Knowledge discovery and data mining, pages 337–341. ACM.

[7] R. U. Kiran and P. K. Reddy.2011. Novel Techniques to Reduce Search Space in Multiple Minimum Supports Based Frequent Pattern Mining Algorithms. ACM 978-1-4503-0528-0/11/0003.

[8] Z. Zheng, R. Kohavi, and L. Mason. Real world performance of association rule algorithms. In KDD '01: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, pages 401–406. ACM, 2001.

[9] L. Zhou and S. Yau. Association rule and quantitative association rule mining among infrequent items. International Workshop on Multimedia Data Mining, 2007.

[10] R. U. Kiran and P. K. Reddy. Mining rare association rules in the datasets with widely varying items' frequencies. In DASFAA (1), pages 49–62, 2010.

[11] M. Walaa, H. Ahmed and K. Hoda, "Combined Algorithm for Data Mining using Association rules," Ain Shams of Electrical Engineering J., Vol. 1.No. 1 ISSN: 1687-8582, 2008.

[12] P. Jyothi, V. D. Mytri, "A Fast association rule algorithm based on bitmap computing with multiple minimum supports using max constraints," International of Comp. Sci & Electronics J.,Volume1,Issue 2, IJCSEE,2013.

[13] F.A. Hoque , N. Easmin and K. Rashed, " Frequent pattern mining for multiple minimum supports with support tuning and tree maintenance on incremental database," Research of Information Technology J., 3(2): 79-90 ,2012.