ANALYSING AND FINDING FREQUENT PATTERNS USING MULTIPLE MINIMUM SUPPORT THRESHOLD

M. Sinthuja¹, D. Devikanniga²

¹Department of Computer Science and Engineering, Presidency University, Bangalore, India
²Department of Computer Science and Engineering, Presidency University, Bangalore, India

¹sinthuja@presidencyuniversity.in, ²devikannigha@presidencyuniversity.in

Corresponding Author: M. Sinthuja
Email: sinthuja@presidencyuniversity.in

https://doi.org/10.26782/jmcms.2020.07.00015

Abstract

Data mining is the process of discovering interesting patterns from the transactional database. In the past decade, numerous techniques have been proposed for mining frequent patterns using single minimum support threshold for all items from the transactional database which results in "rare item issue". While fixing the minimum support to higher level, it results frequent patterns where rare item are missed. While fixing the minimum support to lower level, it results in too many frequent patterns which is known as combinatorial explosion. To confront the rare item problem, an effort has been made in the literature to find frequent patterns with "multiple minimum supports thresholds". In this approach, minimum item support (MIS) is given to each item for mining frequent patterns. In this article, comparative analysis is done between MISFP-Growth and MISLP-Growth algorithm for mining frequent patterns using multiple minimum support threshold. In MISLP-Growth algorithm array based structure is adopted which is the major advantage and in MISFP-Growth algorithm pointer based structure is adopted which is the disadvantage. For this, the experiments are conducted using benchmark databases to find the efficient algorithm. From the results produced by these algorithms, it is found that the MISLP-Growth algorithm outperforms MISFP-Growth algorithm for all the databases in the criteria of consumption of runtime and memory.

Keywords : Data Mining, Frequent Patterns, LP-Growth, Minimum support, Multiple Minimum Support

I. Introduction

Motivation of association rule mining is to solve the issue of Market Basket Analysis. Two most important steps of association rule mining is to mine frequent
patterns and the second is generated by the frequent mode to meet the conditions of the association rules. Numerous algorithms have been developed for mining frequent patterns [I] [II] [IV]. Agarwal is the first author to introduce association rule and gave the algorithm AIS, but the performance of the algorithm is poor. Project lattice theory is established in the year 1994, by utilizing the above algorithm Apriori algorithm is proposed. Apriori is the classic algorithm for finding frequent patterns which is a single minimum support based frequent pattern mining technique [V] [VI] [VII] [VIII]. Pruning and candidate generation is expensive in this algorithm. They tried to discover all rare itemsets, but most time is spent in searching for non-rare itemsets. To emphasize the “rare item problem”, effort have been made in the literature to discover frequent patterns using “multiple minimum support threshold” [III] [XI] [XII]. Apriori like algorithm known as Multiple Support Apriori (MSApriori) is proposed to mine frequent patterns using “multiple minimum support threshold” [II] [XIII]. MSApriori algorithm adopts candidate generation and test approach to explore the entire set of frequent patterns. Sorted closure property is used to reduce the search space and computational cost.

MSApriori algorithm also suffers from the same performance problem as Apriori algorithm face. The problem includes the exploration of huge number of candidate patterns and also the multiple scans of the database [IX]. Conditional Frequent Pattern Growth (CFP-Growth) algorithm which is an extension of FP-Growth algorithm has been proposed for exploring frequent patterns using multiple minimum support patterns [IV] [X]. It employs depth first search approach to generate frequent patterns. It accepts multiple item support values and transactional database as input. By using this multiple item support, CFP-Growth algorithm explores entire frequent patterns.

Multiple Minimum Support threshold for mining frequent itemset: Consider the group of items \( I = \{i_1,...,i_n\} \) an itemset \( X = \{i_1...i_d\} \), and the minimum support threshold (MIS) of \( X \) is stated as: \( \text{MIS}(X) = \text{MIN}\{\text{MIS}(I_1), \text{MIS}(I_2)\ldots\text{MIS}(I_n)\} \). Consider the itemset \( X = \{a,b,c\} \) has an actual support of 8% from a given database. Assume that the MIS value of every item is given as: \( \text{MIS}(a) = 5\% \), \( \text{MIS}(b) = 10\% \), \( \text{MIS}(c) = 15\% \) and the actual support of each items are given as: \( \text{sup}(a) = 10\% \), \( \text{sup}(b) = 9\% \), \( \text{sup}(c) = 11\% \) than the MIS of the itemset \( K \) can be defined as: \( \text{MIS}(XX) = \text{MIN}\{\text{MIS}(aa) = 5\%, \text{MIS}(bb) = 10\%, \text{MIS}(cc) = 15\%\} = 5\% \). Thus, frequent itemset is \( X \) with support 8%, which transcends MIS of \( X \) which has 5%. This is known as downward closure property of multiple item support [XI].

### III. Analysing and Finding Frequent Patterns Using Multiple Minimum Support Threshold

In this comparative research analysis paper, comparison is made between MISIFP-Growth (Frequent Pattern based on Multiple Item Support) and MISLP-Growth (Linear Prefix tree based on Multiple Item Support) algorithms to mine frequent patterns with multiple minimum support thresholds. The bottom-up tree based algorithm, MISFP-growth, is revisited to be self-contained.
MISFP-Growth Algorithm

The MISFP-Growth algorithm is the extension of FP-Growth algorithm. We start by explaining a motivating example that will be used in the presentation of the algorithms.

Consider a transaction database D shown in Table 1 which contains transaction id, items bought and the ordered items. For each item, multiple minimum support is given in the Table 2. Last column of the Table 2 contains the actual support count of each item which is in the database D. In the right most column of the Table 1, items that is present in the transaction database is sorted in descending order based on the minimum support thresholds. To mine frequent patterns MISFP-Growth algorithm is composed of two steps,
1. Building of pattern growth tree
2. Generation of frequent patterns from the tree

Discard property is used in this algorithm where the items which has support count lower than minimum of MIS (MIN-MIS) is discarded and it is not used. Let us go through our motivating example to explain this property. From the second column of Table 2 it is found that the least minimum support threshold is 2. By adopting pattern growth tree called MISFP-Growth (Frequent Pattern based on Multiple Item Support) frequent patterns can be mined by considering multiple support thresholds. This tree is established by scanning all the transactions that exist in the transaction database. The steps of MISFP-Growth algorithm is as follows.

Find the support of each item present in the transaction database D as displayed in Table 2. Discover the least minimum support threshold among all minimum support thresholds. MIN-MIS=2. Scan the database and build MISFP-Growth tree with respect to the right column of the Table 1. Construction of MISFP-tree begins with root node named “Root”. Every transaction which have support greater than or equal to 2 is infused into the tree. From the database the first transaction which contains the itemset {Laptop, Monitor, USB, Cable} is inserted into MISFP-tree.

Table 1: Transaction Database

| TID | Item bought          | Ordered item        |
|-----|----------------------|---------------------|
| 100 | Cable, Monitor, USB, Laptop | Laptop, Monitor, USB, Cable |
| 200 | Mouse, Monitor, USB, Laptop, Tab | Laptop, Monitor, USB, Mouse, Tab |

Copyright reserved © J. Mech. Cont.& Math. Sci.
M. Sinthuja et al
Table 2: Prioritize the items

| Items      | MIS | Actual Support |
|------------|-----|----------------|
| {Laptop}   | 4   | 4              |
| {Monitor}  | 4   | 4              |
| {USB}      | 4   | 3              |
| {CD}       | 3   | 3              |
| {Mouse}    | 3   | 2              |
| {Cable}    | 2   | 1              |
| {Tab}      | 2   | 1              |
|            | 2   | 1              |
As it is the first transaction inserted into the tree, the count of each item is stated as 1. By using pointer connect the nodes which have similar items from head of node link of header table as shown in Fig. 1. Second transaction contains the itemset \{Laptop, Monitor, USB, Mouse, Tab\}.

**Fig. 1:** Pruning item “Cable”
Fig. 2: Pruning item “Tab” and “Headset”

Fig. 3: Complete MISFP-tree
As this transaction shares the prefix {Laptop, Monitor, USB} with the previous transaction the count of each node is incremented as 2 and a new node is created for items Mouse and Tab from the item USB and its count is stated as 1. Repeat the same procedure for all other transactions. By using discarding property any item that have support lower than minimum MIS (Multiple Item Support) is removed and is not used anymore. With reference to Fig. 1 the item “Cable” is removed as its support is lower than minimum MIS. In the same way item “Tab and Headset” is discarded as its support is lower than minimum MIS as shown in Figure 2. Complete MISIFP-Growth is shown in Fig. 3. Bottom up approach is used for mining frequent patterns. Mined frequent patterns using multiple minimum support is shown in Table 3.

Table 3: Mined Frequent Patterns

| Items   | MIS | Conditional pattern base | Conditional MISIFP-tree | Frequent patterns generated |
|---------|-----|--------------------------|-------------------------|-----------------------------|
| {Laptop}| 4   | -                        | -                       | No                          |
| {Monitor}| 4  | {Laptop: 3}              | -                       | No                          |
| {USB}   | 3   | {Laptop:3 Monitor:3}     | { Laptop, Monitor: 3}   | { Laptop, Monitor, USB:3}   |
| {CD}    | 3   | {Laptop:1 Monitor:1 USB:1} | -                       | No                          |
| {Mouse} | 3   | {Laptop:1} {CD:1}        | { Laptop:1 Monitor:1 USB:1} | -                       |
| {Cable} | 2   | -                        | -                       | -                           |
| {Tab}   | 2   | -                        | -                       | -                           |
| {CD}    | 2   | -                        | -                       | No                          |

MISLP-Growth algorithm

An algorithm MISLP-Growth is the extension of LP-Growth algorithm. Transaction database used in MISIFP-Growth algorithm is used for MISLP-Growth algorithm. Sorting the database and finding the least minimum support is similar to that of the MISFP-Growth algorithm. When comparing the structure of both
algorithms MISFP-Growth used pointer based structure where the generation of node is required for each item.

On the contrary, MISLP-Growth algorithm used array based structure which adopts linear structure which is the major advantage of this algorithm. After sorting the database and finding the MIN-MIS start inserting the transactions into the tree by scanning the database. Insert the first transaction which contains the itemset \{Laptop, Monitor, USB, Cable\} into the tree by creating five nodes from the root node which includes the header node. The count of each node is stated as 1 and node link as "null" as it is the first inserted itemset into the tree. Flag is stated as "false" as no branch is created from the particular node. 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 shown in Fig. 4.

![Header List](image)

**Fig. 4: Pruning item “Cable”**
Second transaction’s itemset \{Laptop, Monitor, USB, Mouse and Tab\} shares its prefix path with the first transaction which have the items \{Laptop and USB\} and its count is incremented to 2. New node of three nodes are generated form the item “USB” which includes the header table and insert those items with count 1. Updated the node links and flag value. The same process is repeated for all other transactions.
Disposal property is used to discard the items which have lower minimum MIS (Multiple Item Support). With reference to the Fig. 4 item “Cable” is removed as its support is lower than the minimum MIS. In the same way remove the item “Tab” and “Headset” as shown in the Fig. 5. The complete MISLP-tree is shown in Fig. 6. Bottom up approach is adopted for mining frequent patterns. Complete frequent itemset is mined by fixing multiple minimum support threshold to each item in the transaction database as shown in Table 4.

Table 4: Mined Frequent Patterns

| Items     | MIS | Conditional pattern base | Conditional MISIFP-tree | Frequent patterns generated |
|-----------|-----|--------------------------|--------------------------|----------------------------|
| \{Laptop\} | 4   | -                        | -                        | No                         |
| \{Monitor\} | 4    | \{Laptop: 3\}           | -                        | No                         |
| \{USB\}   | 3   | \{Laptop :3 Monitor: 3 \} | \{ Laptop, Monitor: 3 \} | \{Laptop, Monitor, USB:3\} |
| \{CD\}    | 3   | \{Laptop:1 Monitor: 1 USB: 1\} | -                        | No                         |
| \{Mouse\} | 3   | \{Laptop:1\}\{CD:1\}    | -                        | No                         |
| \{Cable\} | 2   | -                        | -                        | -                          |
| \{Tab\}   | 2   | -                        | -                        | -                          |
| \{Headset\} | 2    | -                        | -                        | No                         |

VI. Performance Evaluation

In this section, algorithms of MISFP-Growth and MISLP-Growth are compared to mine frequent patterns using multiple minimum support thresholds in the criteria of runtime and consumption of memory.

Experimental environment and datasets

Two datasets are used for the experimental evaluation of mining frequent patterns. In this experiment, the performance is evaluated between MISFP-Growth and MISLP-Growth algorithm. The experiment is conducted on Intel® corei3™ processor.
CPU, 2.13 GHz, and 2GB of RAM computer. To evaluate the performance, implementation is done in Java. The Characteristics of database is shown in Table 5.

| Datasets  | Size (MB) | # of distinct items | Average Transaction length | # of Transactions |
|-----------|-----------|---------------------|---------------------------|------------------|
| Retail    | 4.07      | 21387               | 10.3(76)                  | 88162            |
| T10I4D100K | 3.83      | 870                 | 10.1                      | 100,000          |

Consumption of Runtime

In this subsection, runtime is calculated. Runtime is the time between the input and output. The tree structure algorithms of MISFP-Growth and MISLP-Growth are compared. Both the algorithms mines frequent patterns using multiple minimum support. From the Fig. 7 x-axis represents minimum support and y-axis represents time in Seconds. For the database Retail the runtime of the MISFP-Growth algorithm is higher than MISLP-Growth algorithm for all α values. As α increases, the number of frequent itemsets generated decreases. As a result runtime of the MISFP-Growth and MISLP-Growth algorithms decreases.

Fig. 7: Runtime of MISLP-Growth and MISFP-Growth algorithm

MISLP-Growth algorithm is better than MISFP-Growth algorithm. Since it apply pruning and merging which is time consuming, the structure of the algorithm which is an array based form decreases the runtime in mining frequent itemsets. Runtime of MISLP-Growth algorithm are very close to one another and they are decreased from 1 to 0.7 for mining frequent itemset using multiple minimum support threshold.

Copyright reserved © J. Mech. Cont. & Math. Sci.
M. Sinthuja et al
With reference to Fig. 8, for the database T10I4D100K it is clearly shown that the consumption of time for MISLP-Growth algorithm is lower when compared to the MISFP-Growth algorithm. In every value of $\alpha$ MISLP-Growth algorithm performs better than MISFP-Growth algorithm. Runtime value of MISLP-Growth algorithm seems have same values, but actually these values are different and they decrease from 18 to 8 (they are close to each other) for mining frequent itemsets using multiple minimum support threshold.

![Fig. 8: Runtime of MISLP-Growth and MISFP-Growth algorithm](image)

In MISFP-Growth algorithm, for the construction of FP-tree node by node generation is required which requires more time. But in MISLP-Growth algorithm array structure is used which explores multiple nodes within a node. Usage of pointer is reduced as it adopts array structure. The format of the tree increases the speed of the item traversal. Thus, the runtime of MISLP-Growth is lower when compared to MISFP-Growth algorithm.

**Consumption of Memory**

In this subsection, algorithms of MISFP-Growth and MISLP-Growth are compared to mine frequent patterns using multiple minimum support. From the graph longitudinal axis shows the memory in MB and latitudinal axis shows the different minimum support threshold.

From the Fig. 9 and Fig. 10, it is observed that the MISLP-Growth algorithm consumes lower memory when compared to MISFP-Growth algorithm for the database Retail and T10I4D100K. For the database Retail MISFP-Growth algorithm consumes more memory in most of the cases. The reason is node by node generation is needed which required more pointers. Memory required to store pointers consumes

*Copyright reserved © J. Mech. Cont. & Math. Sci.*

*M. Sinthuja et al*
more memory. It shows MISLP-Growth algorithm consumes lower memory than MISFP-Growth algorithm.

![Graph showing memory consumption comparison between MISLP-Growth and MISFP-Growth algorithms](image)

**Fig. 9:** Consumption of memory of MISLP-Growth and MISFP-Growth algorithm

Consumption of memory for the database T10I4D100K shows that the MISLP Growth algorithm lower memory when compared to MISFP-Growth algorithm. The reason is usage of pointer is limited as the algorithm adopts linear structure. No pointer is required to link the internal nodes. Thus the MISLP-Growth algorithm lessens the memory space when compared to the MISFP-Growth algorithm in mining frequent itemsets using multiple minimum support threshold.
V. Conclusion

The frequent itemset mining algorithms establish the frequent itemsets from a database. But they face many challenges such as accessing data multiple times, response time, huge sizes of database, single support threshold, dynamicity nature of the databases, etc. Single support threshold that does not allow user to specify support threshold according to the nature of the items. For this, multiple minimum support is assigned to each item and it is found that frequent itemset which involves rare items can be extracted in more efficient manner.

In this paper, analysis and comparison is made between MISFP-Growth and MISLP-Growth algorithms to extract frequent patterns which involves rare items. The structure of the MISLP-Growth algorithm is the major advantage of this algorithm as it adopts linear structure which is array based form. That is single node is generated for each transaction which contains multiple internal nodes. But, in MISFP-Growth algorithm node by node generation is required for each item in a transaction which is the major disadvantage. The Performance of these algorithms is evaluated by conducting experiments on datasets such as Retail and T10I4D100K. The results shows that, MISLP-Growth algorithm prunes frequent patterns in more efficient manner than MISFP-Growth algorithm in terms of runtime and usage of memory.
References

I. Agrawal, R., and Srikant, R. (1994) ‘Fast algorithms for mining association rules in large databases’, In Proceedings of the 20th International Conference on Very Large Data Bases, VLDB ’94, pp.487–499.

II. Borah, A., and Nath, B. (2019) ‘Rare Pattern Mining: Challenges and Future Perspectives.’ Complex Intell. Syst. 5, pp. 1-23.

III. Chee, C., Jaafar, J., Aziz, I.A. (2019)’ Algorithm for Frequent Itemset Mining: A Literature Review’ Artificial Intelligence, 52, pp.2603-2621.

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

V. Han, J., Pei, J., Yin, and Mao, R, (2004) ‘Mining frequent patterns without candidate generation: A frequent pattern mining tree Approach, Data Mining in Knowledge Discovery 8(1): pp.53–87.

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

VII. Hu, Y.H., and Chen, Y, (2006) ‘Mining association rules with multiple minimum supports: a new mining algorithm and a support tuning mechanism’, Decis. Support Syst., 42(1): pp.1–24.

VIII. Kiran, and Reddy, P.K. (2010) ‘Mining rare association rules in the datasets with widely varying items’ frequencies’, In DASFAA (1), pp. 49–62.

IX. Liu, B., Hsu, W., and Ma. Y. (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, ACM, pages 337–341.

X. Sinthuja, M, Puviarasan, N, and Aruna, P, (2018) ‘Proposed Improved FP-Growth Algorithm with Multiple Minimum Support Threshold Value (MISIFP-Growth) For Mining Frequent Itemset’, International Journal of Research in Advent Technology, Vol.6, May, pp.471-476.

XI. Sinthuja, M, Puviarasan, N, and Aruna, P, (2018) ‘Mining frequent Itemsets Using Top Down Approach Based on Linear Prefix tree’, Springer, Lecture Tabs on Data Engineering and Communications Technologies, Vol.(15), September, pp.23-32.
XII. Sinthuja, M, Puviarasan, N, and Aruna, P, (2018) ‘Geo Map Visualization for Frequent Purchaser in Online Shopping Database Using an Algorithm LP-Growth for Mining Closed Frequent Itemsets’, Elsevier, procdia computer science, Vol.132, pp.1512-1522.

XIII. Sinthuja, M. Puviarasan, N. and Aruna, P, (2019) Frequent Itemset Mining using LP-Growth algorithm based on Multiple Minimum Support Threshold Value (MIS-LP-Growth), Journal of Computational and Theoretical Nanoscience, Volume 16, No(4), pp. 1365-1372(8).