MMP-TREE FOR SEQUENTIAL PATTERN MINING WITH MULTIPLE MINIMUM SUPPORTS IN PROGRESSIVE DATABASES

K.M.V.Madan Kumar[1], P.V.S.Srinivas[2] and C.Raghavendra Rao[3]

[1] Research Scholar, CMJ University, Shillong, Meghalaya, India
Assoc Professor, TKR College of Engg &Tech, Hyderabad, AP, India
[2] Professor, Geethanjali College of Engineering & Technology, Hyderabad, India
[3] Professor, Dept. of CIS, University of Hyderabad, Hyderabad, India

ABSTRACT

Although there may be lot of research work done on sequential pattern mining in static, incremental, progressive databases, the previous work do not fully concentrating on support issues. Most of the previous approaches set a single minimum support threshold for all the items or item sets. But in real world applications different items may have different support threshold to describe whether a given item or item set is a frequent item set. This means each item will contain its own support threshold depends upon various issues like cost of item, environmental factors etc. In this work we proposed a new approach which can be applied on any algorithm independent of that whether the particular algorithm may or may not use the process of generating the candidate sets for identifying the frequent item sets. The proposed algorithm will use the concept of "percentage of participation" instead of occurrence frequency for every possible combination of items or item sets. The concept of percentage of participation will be calculated based on the minimum support threshold for each item set. Our algorithm MS-PISA , which stands for Multiple Support Progressive mIning Of Sequential pAtterns , which discovers sequential patterns in by considering different multiple minimum support threshold values for every possible combinations of item or item sets.

KEYWORDS

Multiple minimum supports, Progressive sequential pattern, Percentage of participation

1. INTRODUCTION

Sequential Pattern mining is one of the most important research issues in data mining. Which was first introduced by Agarwal and Srikanth [2] and can be described as follows: we are given a set of data sequences which will be used as input data. Each sequence consists of a list of ordered item sets containing a set of different items. The sequential pattern mining finds all Sub sequences with frequencies lower than min support which is given by user. After that there have been a significant research work was done by various researchers on sequential pattern mining and defined number of algorithms not only for static data base [2],[3] Where data do not change over time but also for incremental data bases[6] where there will be new data arriving as the time goes by. In addition to above researchers some other researchers derived other kind of sequential patterns like closed sequential patterns. Constraint sequential pattern, maximal sequential patterns, spatio temporal sequential patterns. Sequential pattern on specific type of data, on stream data.

DOI : 10.5121/ijcsea.2012.2506

The sequential patterns mining in progressive data bases will be having a vide use in many applications. For example in mobile applications, share market. Moreover the stock price changes of the company before five years ago may have very little influence on the quotes of other stocks. To remedy the problem of sequential pattern mining in progressive databases, Haung.et.al [6] proposed an algorithm **PISA** which takes the concept of period of Interest (POI) but Haung.et.al[6] also considered only uniform min.sup like all the previous researchers.

But considering uniform min.sup, implicitly assumes that all items in the data base have similar frequency. However some items may appear very frequently in the data base while other rarely appears. Under such circumstances, if we set the value of min support too high we will not find those rules involving rare items in the data base. On the other hand if we set that value too low, it will generate huge amounts of meaningless patterns. Therefore liue .et.al [4] first address this and propose the concept of multiple minimum supports (MMS in Short)in association rule mining. In this study we first extend the definition of sequential pattern by considering the concept of MMS, which allows users to specify multiple min.sup for each item. Items with low frequencies will be specified with lower minimum support and pattern involving these items can be easily retrieved for further decision support.

But the major problem on frequent pattern mining with MMS is that the *downward closure property* no longer holds in the mining process, which means that a super-pattern of an infrequent pattern might be frequent pattern. To effectively reduce the search space in a level –wise methods, Liu.et.al proposes the sorted closure property, where all item in data sets are sorted in ascending order by their MIS values. The sorted closure property however is invalid in sequential pattern mining since the order in the data sequences cannot be altered. Therefore to discover complete set of sequential patterns with MMS is not straight forward. Based on the new definitions of sequential patterns with MMS, we first proposed the concept called *Percentage of participation (POP)* where POP is the percentage of participation of the item or item set with respect to the no of sequences in the POI i.e. |Db| and minimum item support (MIS).

The structure of this paper as follows. In section 2, we first review the concept of **PISA** as the basis of our approach. Section 3 introduces the definition of the progressive sequential pattern mining with multiple minimum supports (MMS) by involving the Percentage of participation (POP).Our algorithm called **MS-PISA** will be discussed in section 4 and we have conclusion in section 5.

2. RELATED WORK

Haung.et.al proposed algorithm which works for mining of sequential patterns in progressive data bases. The input of progressive sequential pattern mining is a user specified length of POI and user-defined minimum support threshold. POI is a sliding window protocol whose length is a user specified time interval continuously advances as the time advances. The sequences having elements whose time stamps are within the POI will be considered to contribute for |Db| for current sequential patterns (where |Db| is the number of sequences in the POI). The sequences having elements with time stamps older than the POI will be considered as absolute and removed. In this they consider an element which is set of items appeared in a sequence with the time stamp denoted by p, q. When the sequence database is denoted by Db then a sequence in a sequence database is a set of elements ordered by time stamp increasingly. They also introduced a time interval denoted by [p, q] to represent the time interval between time stamp p and q. Therefore Dbp,q is defined as a subset of the databases Db containing the elements of sequences from time stamp p to time stamp q. They also aimed finding the sequential patterns containing no repeated elements and having more than one element. In the beginning, the mining algorithm receives new data of different timestamps with a user-defined minimum support threshold and the length of

POI. For each POI, the mining algorithm progressively updates the sequences in the database. Then, the algorithm outputs frequent sequential patterns which are qualified by the uniform minimum support threshold. Note that the mining algorithm should prune away obsolete data from the sequence database, delete obsolete sequential patterns rapidly, and update |Db| with the number of sequences which have elements in the urrent POI. The process will be continuously executed until there is no more newly arriving data. To solve the progressive sequential pattern mining algorithm, Haung.et.al proposed a progressive mining algorithm PISA. Where PISA maintains a PS-tree to keep the information of the progressive database and up-to-date sequential patterns in each POI. The nodes in PS-tree can be divided into two different types. They are root node and common nodes. Root node is the root of PS-tree containing nothing but a list of common nodes as its children. Each common node stores two information, say node label and a sequence list, as shown in the top-left side of Fig. 2. The label is the same as the element in a sequence. The sequence list stores a list of sequence IDs to represent the sequences containing this element. Each sequence ID in the sequence list is marked by a corresponding timestamp. Whenever there are a series of elements appearing in the same sequence, there will be a series of nodes labeled by each element, respectively, with the same sequence IDs in their sequence lists. Then, the first node will be connected to the root node and the second node representing the following element will be connected to the first node. The other nodes will be connected analogously. Note that, in such a way, the path from root node to any other node will represent the candidate sequential pattern appearing in this sequence. The appearing timestamp for each candidate sequential pattern will be marked in the node labeled by the last element. If there is another sequence having the same pattern, the sequence ID will be inserted into the sequence lists of the same nodes labeled by these elements on the path. On the other hand, if an element appearing in a sequence is obsolete, the corresponding sequence ID will be removed from the sequence list of the node. In addition, if a node has no sequence in the sequence list, it will be pruned away from PS-tree.

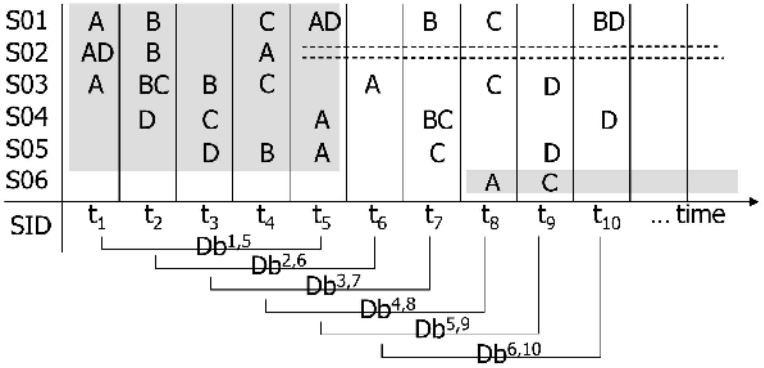

Figure 1

For the above example shown in figure1 the algorithm PISA constructs the candidate sets as follows which is shown in figure2.

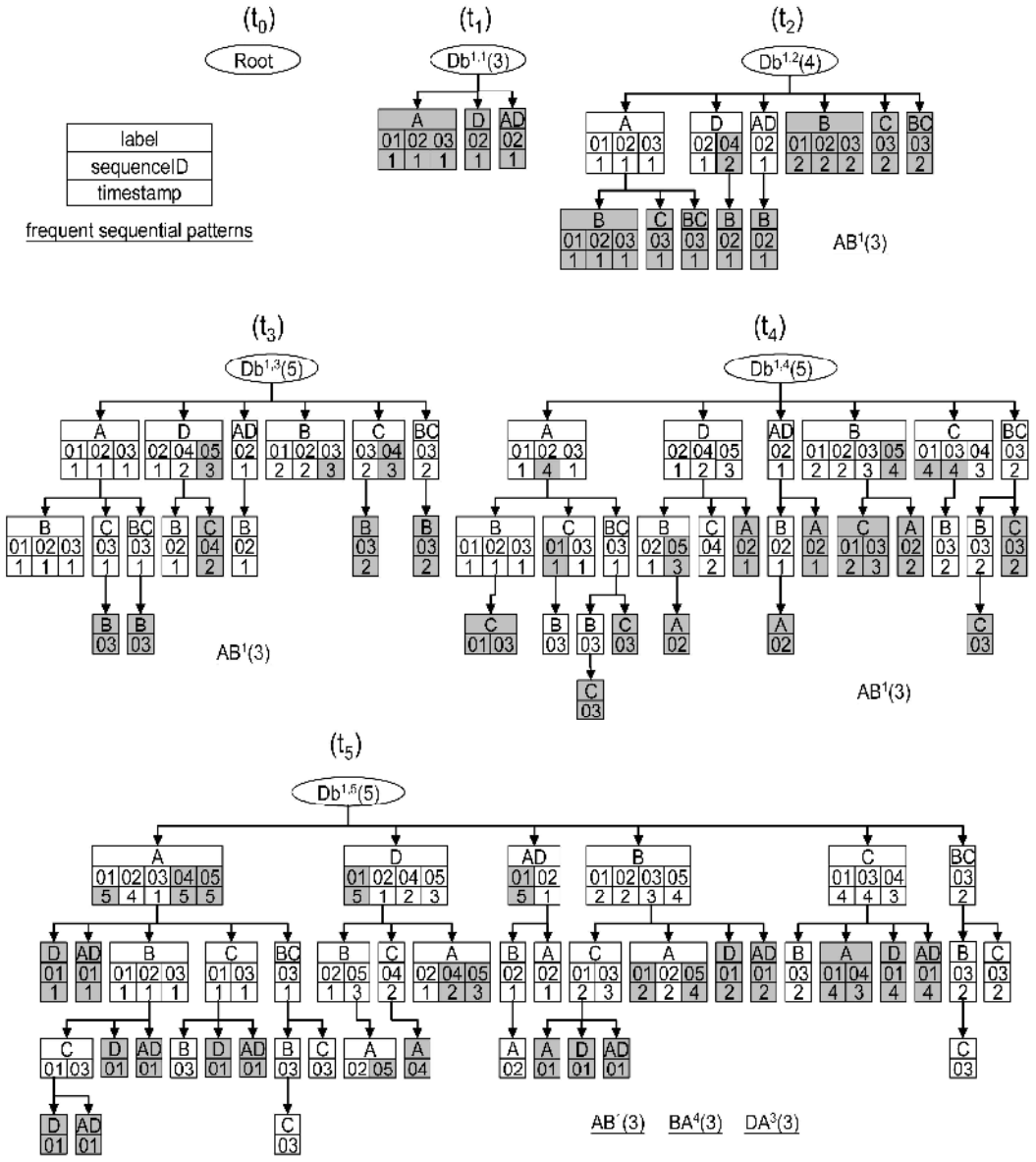

Figure 2

Before this many researchers have developed various methods to find frequent sequential patterns with a static database. Apriori All [2], GSP [3] are the milestones of sequential pattern mining algorithms based on traditional association rules mining technique. SPADE [5], illustrated by Zaki, systematically searched the sequence lattice spanned by the subsequence relation. Han et al. and Pie et al. brought up Free Span and Prefix Span, which found sequential patterns by constructing sub databases of the entire database. Zhang et al. proposed two algorithms GSP+ and MFS+ based on static algorithms GSP and MFS (also derived by the same authors).But as discussed early in section1 all the researchers considered uniform min.sup to describe whether the sequence is frequent or not.

3. PROBLEM DEFINITION

In this section, we formally give definitions used in the discovery of sequential pattern with MMS. Let I denote the set of items in the database, and a subset of I is called an item set. A customer's data-sequence is an ordered list of items with time stamps. Therefore, a sequence, say α, can be represented as $<(a_1: t_1), (a_2: t_2), (a_3: t_3), \ldots, (a_n: t_n)>$, where a_j is an item, and t_j stands for the time when a_j occurs, $1 \leq j \leq n$, and $tj-1 \leq tj$ for $2 \leq j \leq n$.
If several items occur at the same time in t_j for 2 j the sequence, they are ordered alphabetically.

Definition 1. Given an item set $I_q = (i_1, i_2 \ldots i_m)$, we say item set I_q occurs in α if integers $1 \leq k_1 < k_2 < \ldots < k_m \leq n$ exist such that, $i_1 = a_{k1}$, $i_2 = a_{k2}, \ldots, i_m = a_{km}$ and $t_{k1} = t_{k2} = \ldots = t_{km}$. We refer to k_1 and t_{k1} jointly as the position and the time that I_q occurs in α, respectively.

Definition 2. Let $\beta = <I_1, I_2 \ldots I_s>$ (I_q subset or equal to I for $1 \leq q \leq s$) be a sequence of item set. Assume that each I_q in β occurs in α. Then we say sequence β occurs in α, or is a subsequence of α if $t_{I1} < t_{I2} < \ldots < t_{Is}$, where t_{Iq} ($1 \leq q \leq s$) is the time, at which I_q occurs in α.

Definition 3. A sequence database S is formed by a set of records $<sid, s>$, where *sid* is the identifier of this data-sequence and s is a data-sequence. For a given sequence β, the support count of sequence β in S are defined as follows:

$$supp(\beta) = |\{(sid, s) | (sid, s) \in S \wedge \beta \text{ is a subsequence in } s\}|$$

The following definitions are related to the concept of MMS. In this model, the definition of the minimum support is changed. Each item in the database can have its *minsup*, which is expressed in terms of *minimum item support* (MIS). In other words, users can specify different MIS values for different items.

Definition 4: Let MIS (i) denote the MIS value of item i (i subset or equal to I). Given an item set $I_q = (i_1, i_2, \ldots, i_m)$, the MIS value of item set $I_q = (i_1, i_2, \ldots, i_m)$ ($1 \leq k \leq m$), denoted as MIS(I_q), is equal to:

$$\min [\text{MIS}(i_1), \text{MIS}(i_2), \ldots, \text{MIS}(i_m)]$$

Definition 5: Given a sequence $\beta = <I_1 I_2 \ldots I_s>$
(I_q subset or equal to I for $1 \leq q \leq s$), the minimum support threshold of β, denoted as MIS (β), is equal to:

$$\min [\text{MIS}(I_1), \text{MIS}(I_2), \ldots, \text{MIS}(I_s)]$$

Definition 6: Given a sequence database S and a sequence β, we call β is frequent in S or β is a sequential pattern in S if $supp(\beta) \geq \text{MIS}(\beta)$.

Definition 7: Progressive sequential pattern mining problem. "Given a user-specified length of POI and a user- defined minimum support threshold, find the complete set of frequent subsequences whose occurrence frequencies are greater than or equal to the minimum support times the number of sequences in the recent POI of a progressive database."

Definition 8: Let I be an item or item set and MIS (I) be the minimum item support threshold of item I. $|Db^{p,q}|$ is the number of sequences In the POI. Timestamps between p ,q.

$$\text{Percentage Of Participation(POP)} = 100/ |Db^{p,q}| * MIS(I)$$

With the provision of different minimum item support thresholds for different items, user can effectively express different support requirements for different data-sequences. MMS allow users to have higher minimum supports for sequences involving high frequent items, and also allows us to have lower minimum supports for sequences involving rare items. Given a sequence Database S and a set of MIS values for all items in S, we discover all sequential patterns that satisfy MIS (*β).*

4. THE PROPOSED ALGORITHM

Progressive Mining of sequential patterns with multiple minimum supports

For mining of sequential patterns in progressive databases with multiple minimum supports, we propose another algorithm called MS-PISA. The algorithm maintains a tree based data structure called MMP-tree which will store the knowledge of the database, where the database is progressively incremented as the new transactions has been taken place. The tree also maintains the percentage of participation of each element whenever the element is formed. Here we will first discuss about MMP-tree in section 4.1.Next we explained in detail about MS-PISA in section 4.2.In section 4.3 we have taken an example database and discussed very clearly how the algorithm MS-PISA works on the example.

4.1 MMP-tree

Here the MMP-tree will act as a basic kernel of MS-PISA. It not only stores the knowledge of all the sequences in the database which progressively updates and also helps the algorithm to gather frequent sequential patterns according to the multiple minimum supports in each POI. The nodes in MMP-tree can be categorized in to two different categories. One is the root node and another is common node. Root node is root of the MMP tree which does not stores any other information but attached to common nodes as its children. Each common node gathers the knowledge about three issues, one node label, second the sequence list of itemset, third the percentage of participation of the itemset according to the minimum support of itemset.

Here the label is the same as the element in a sequence. The sequence list is the ID of the sequence that contains the element to represent the sequence. The sequence ID in the sequence list will be attached to the corresponding time stamp to help in deleting the obsolete data. It is worth to mention that only the nodes in the first and the second levels have to maintain the corresponding time stamps for the sequence IDs. The time stamps for the sequence IDs in the nodes below the third level are the same as the time stamps for the same sequence IDs in the second level. Therefore it is not necessary to store the duplicate information. Third along with label, sequence ID the MMP-tree also stores the information about the percentage of participation of the element according to the minimum support of the element. For example if the element is AB then the tree finds out which item in the element is having less user defined minimum support. If A is having less minimum support, then the percentage of participation will be calculated with respect to A by using the formula $POP=100/|Db^{p,q}| * MIS(A)$ where $|Db^{p,q}|$ is no of sequences in the time slot from time stamp p to time stamp q.

When ever in a particular sequence, if the element is appearing in a series wise then series of nodes labeled by each element along with the percentage of participation will be gathered respectively. The sequence ID will be same the sequence list. In the process of forming of the

MMP tree the first node will be connected to the root node with the element label and percentage of participation of the element with respect to minimum support of the element. The second node will be representing the following element along with the percentage of participation of that particular element with respect to the minimum support of that element. The second node will be attached to the first node which is already formed. The other nodes will be formed with corresponding elements and their percentage of participation analogously. Note that in such a way the path from root node to any other node will be representing the candidate sequential pattern appearing in this sequence. The appearing time stamp for each candidate sequential pattern will be marked in the node labeled by last element. If there is another sequence having the same pattern, the sequence ID will be inserted into the sequence lists of the same nodes labeled by these elements on the path along with the percentage of participation of the element. On the other hand, if an element appearing in a sequence is obsolete, the corresponding sequence ID will be removed from the sequence list of the node. When an obsolete node is removed the percentage of participation of the element also will be removed. In addition, if a node has no sequence in the sequence list, it will be pruned way from MMP tree. Thus we can ensure that there are only up-to-date candidate sequential patterns in MMP tree. Now we will consider those elements whose percentage of participation in more than 100% in total as frequent sequential patterns from the above formed candidate sequential patterns, now we will demonstrate how to maintain MMP-Tree in detail in the following section i.e. 1.2.

4.2 Algorithm MS-PISA

The main process in MS-PISA is to progressively update the knowledge of each sequence and accumulate the percentage of participation of each element in the sequence. To construct the candidate sequential pattern in progressive data base, the MS-PISA will use the MMP-Tree. For this, the MMP-Tree mainly stores all the sequences along with the percentage of participation corresponding to the minimum support of element that are falling in the sequences from one POI to another. When receiving the elements at the arriving time stamp, say t+1, MS-PISA constructs the original MMP-Tree of time stamp t in post order (children first, then the node itself) by accumulating the POP and updates the MMP-Tree of time stamp t.

In the process of MMP-Tree, MS-PISA do the following things:

1. Deletes the obsolete elements along with the corresponding percentage of participation from.
2. Update the current sequences with the POP.
3. Calculate the POP of the new arriving elements and inserts them in to MMP-Tree of time stamp t.

The detailed algorithm for POP is shown in the figure 3(a) and for MS-PISA in the figure 3(b), 3(c). In figure 3(b) from line 4 to line 7 MS-PISA collects the elements of all sequences along with the POP of the elements at the running time stamps and constructs the MMP-Tree. Then MS-PISA travels in forward direction to the next time stamp and collets the arriving elements in the sequences along with the POP of the elements. This process continuous until there is no further newly arriving elements in a progressive data base.

The main procedure constructs is used to construct the MMP-Tree of time stamp t and convert it to the new MMP-Tree of time stamp t+1 along with the POP of the elements. The procedure construct is shown in the fig. 3(c) and it was explained in detail with an example in fig.4 the procedure construct MMP-Tree is in post order. That is MS-PISA goes to the children node and deals with them first. After all the children nodes are constructed there MS-PISA constructs to the node itself. The basic concept of procedure construct is to attach each newly arriving element of all the sequences along with percentage of participation of the each arriving element with respect to the minimum support of the element in to the MMP-Tree. There are two kinds of

nodes in MMP-Tree say root node and children nodes. For root node MS-PISA examines all sequences which have new elements and calculate the percentage of participation of the element with the formula 100 / $|Db^{p,q}|$*MIS(el) where el :element. As shown in figure 4(t_1), the sequences S01, S02, S03 consisting of elements A,(AD) and A. The MS-PISA deals with root node first from line 2 to line 11 shown in figure 3(c). If the node is root MS-PISA examines all newly arriving elements in the *eleSet* including the *POP* from line 3 to line 11.MS-PISA has to create all the combinations of candidate elements and calculate the POP of the combinations in newly arriving data. For example if the arriving data contains (ABC), then the combinations will be A, B, C, (AB), (AC), (BC) and (ABC) with their corresponding POP. For each element, if the element already appeared in the last time stamp then MS-PISA checks for the sequence ID was existed in that child node from Line 6 to 9.If it is existed in the list then MS-PISA updates the time stamp of that sequence to new time stamp and calculate the POP of the element by considering the MIS (el).Otherwise MS-PISA creates a new sequence with POP of the elements and inserts the sequence in to the list of the sequence child. If there is no child node is available with the same label, MS-PISA creates new child node with the corresponding sequence ID and calculate the POP of the elements which are present in the node in line 11.

As shown in figure 4(t_2) at node labeled by A ,MS-PISA checks the list of sequences of that node to find if there is any newly arriving element of the sequences ,say 01,02 and 03.The corresponding details are shown in figure 3(c) from line 12 to 27.If the upcoming node is not the root node , MS-PISA first deletes the sequences as obsolete from the list of sequences along with their corresponding POP from line 14 to 15.If there is no ID left in the list MS-PISA prunes way the node along with the POP from parent and goes to next node from line 26 to 27.The MS-PISA the left out sequences in the sequence list from line 16 to line 25 of figure 3(c). If there is a newly arriving element in *eleset*, for all combinations of candidate elements in the arriving data MS-PISA checks whether the element is on the path from root node in line 18.If it is not, that means that there is new candidate sequential pattern. Then MS-PISA follows the procedure from line 19 to 25 and creates new sequences or Childs along with the percentage of participation of elements same as from line 5 to 11.From line 26 to 27 we can note that if there is no sequence in a node that node and it's children along with POP should be deleted.

After processing all the common nodes (which means the process of constructing candidate sequential patterns was completed) then MS-PISA examines the sequence ID's for frequent sequential patterns by considering the total POP of the nodes (including from root to child) should be equal to 100 or more than 100.Now MS-PISA will give the path from root to child as the output from line 28 to 29 of figure 3(c).In this way MS-PISA can find out all up-to-date frequent sequential patterns of time stamp t+1after constructing the original MMP-tree of time stamp t.

The process of calculating the POP will be as follows. As shown in figure 3(a) the algorithm POP will check the elements from newly arriving data. If the element is having more than one item then it will identify the item which is having less MIS value as shown from line 3 to 4 in figure 3(a). Then it will calculate the corresponding POP of the element described from line 6 to8 of figure 3(a).

Consider the example shown in the figure 1 and the MIS values of A, B, C and D are 0.8, 0.5, 0.3 and 0.2 consecutively. The POP of A, B, C and D for 5 sequences are 25, 40, 66, and 100. The figures 4 (a), 4(b), 4(c), 4(d), 4(e), 4(f), 4(g), 4(h) and 4(i) are the MMP-Trees of the example sequence database shown in figure 1. In the same figures we have shown the frequent sequences. The process was explained in detail in the next section.

Algorithm POP (Items, MIS(I))

1. var POP;
2. var I, I_{min};// Items
3. For (all combinations of *items* in the *ele*);
4. If (*No of items* >1);
5. Check and *identify* the item I_{min} having less support;
6. POP (*ele*) =100/$|Db^{p,q}|$*MIS(I_{min}): // $|Db^{p,q}|$ is no of sequences from p to q
7. **else**
8. POP (*ele*)=100/$|Db^{p,q}|$*MIS(I): // $|Db^{p,q}|$ is no of sequences from p to q

END

Figure 3(a)

Algorithm MS-Pisa (POI)

1. var *PS;* // PS-Tree
2. var *current Time*; // timestamp now
3. var *eleSet*; // used to store elements *ele*
4. **while** (there is still new transaction)
5. *eleSet* = read all *ele* at *current Time*;
6. **construct** (*current Time, PT*);
7. *current Time++*

END

Figure 3(b)

Algorithm construct (*current time, PS*)

1. for (each *node* of ***PS*** in post order)
2. **if** (*node* is null)
3. **for** (*ele* of every *seq* in *eleSet*)
4. **fo**r (all combination of elements in the *ele*)
5. **if** *(element*==label of one of *ele.child*)
6. **if** (*seq* is in *node.child.seq_list*)
7. update timestamp of *seq* to *currentTime, POP (ele);*
8. **else**
9. create a new *sequence* with *current Time, POP(ele);*
10. **else**
11. create a new *child* with *element, current Time, POP(ele);*
12. **else for** (every *seq* in the *seq_list*)
13. **if** (*seq.timestamp*<= *currentTime_POI*)
14. delete *seq, POP (el)* from *seq_list* and continue to *next seq*;
15. **if** (there is new *ele* of the *seq* in *eleSet*)
16. **for** (all combination of elements in the *ele*)
18. **if** (element is not in the path from *Root*)
19. **if** (element==label of one of *node.child*)
20. **if** (*seq* is in *child.seq_list*)
21. *child.seq_list.seq.timestamp= seq.timestamp*;
22. **else**
23. create a new sequence with *seq.timestamp, POP (ele*);
24. **else**
25. create a new child with *element, seq.timestamp, POP (ele)*;
28. **if** (*seq_list.Total POP (el)*>=100%)
29. output the *ele* as a sequential pattern;

END

Figure 3(c)

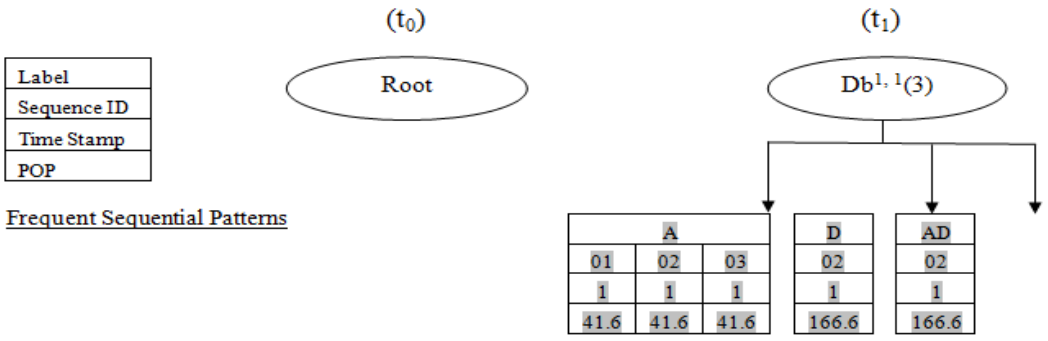

Figure 4(a)

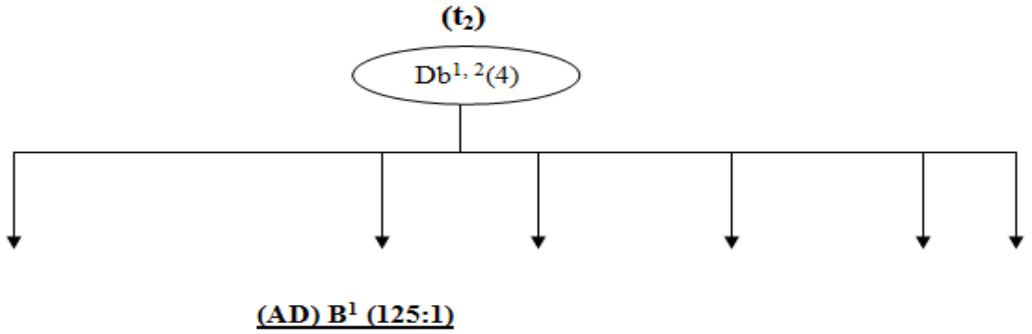

Figure 4(b)

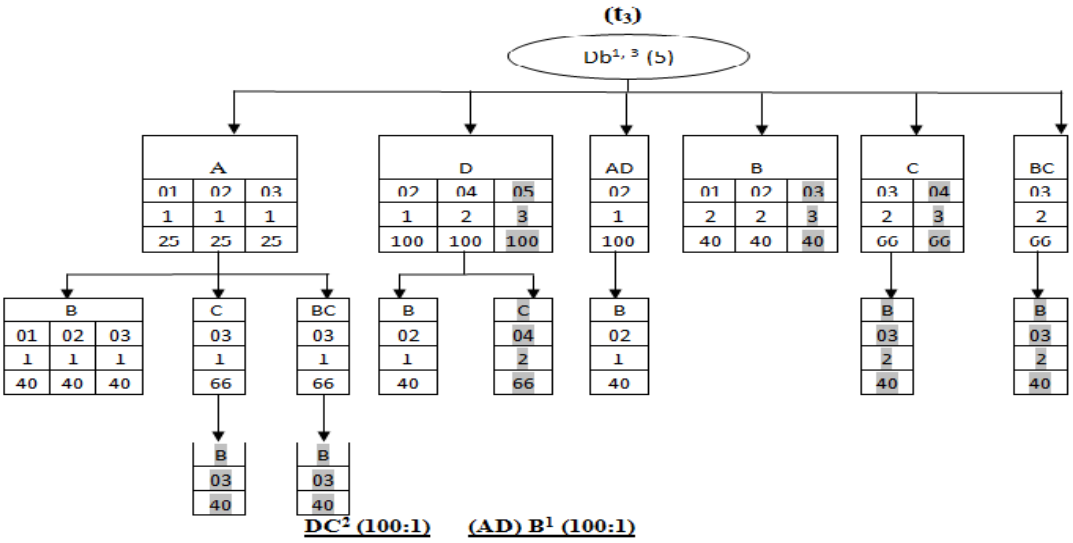

Figure 4(c)

5.RESULT

Our proposed algorithm is very effectively working for test dataset and we have analyzed the test dataset for different parameters like period of interest, execution time etc.

5.1 Impact of period of interest (POI) on execution time

Execution time is the time required to execute all the instructions in the proposed algorithm. Here we can observe that the execution time will increase a little bit with respect to the time taken for identifying the item having minimum MIS value. For this we applied the quick sorting technique .It can be noted that execution time is directly proportional to POI. The reason is that, the increase in POI required more time for process as the no of new elements will be added to the existing one.

Figure 5(a) shows the impact of POI over the execution time.

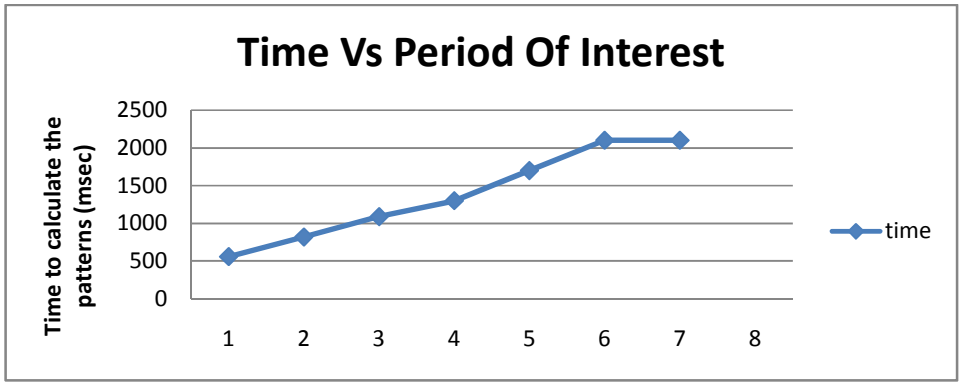

Figure 5(a)

5.2 Impact of POI over number of patterns

Number of patterns is dependent on period of interest a. As from Fig.5(b) we can see that as the period of interest increases, the number of patterns also increases. This is because as the period of interest increases, the algorithm has more items to process and so they give more number of patterns.

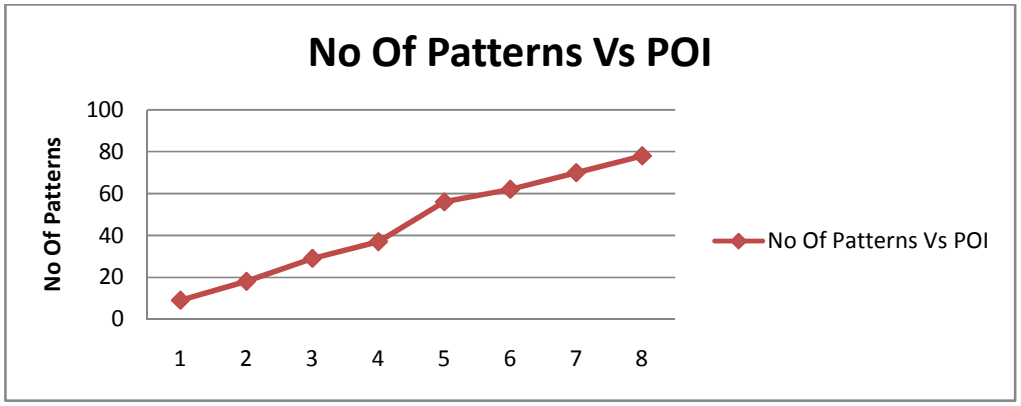

Figure 5(b)

4.3 Comparison between MS-PISA and PISA in terms of time in progressive database

Generally mining frequent sequential patterns with multiple minimum supports will take more time when compared with mining frequent sequences with uniform support. In case of percentage of participation (POP), the time for execution of Ms-PISA will take little bit more time than the PISA. The difference of time between these two will be the time for identifying the item having minimum MIS value from the newly arriving element and accumulating the POP. We can observe the difference in the Figure5(c)

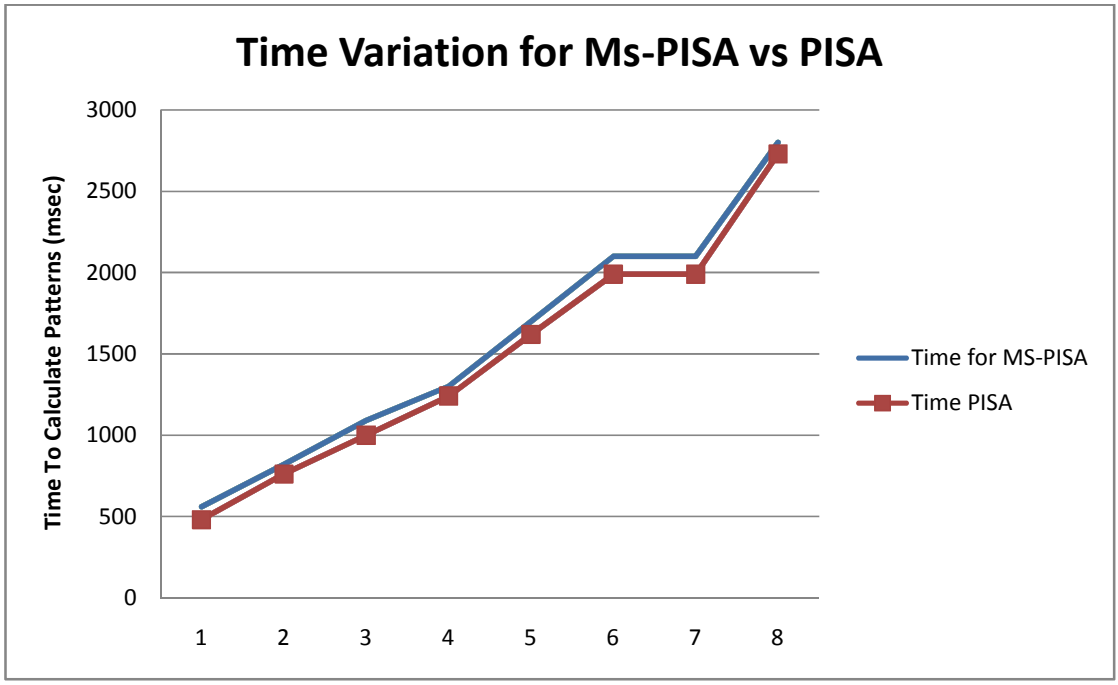

Figure 5(c)

6. CONCLUSION

In the proposed novel work MS PISA algorithm is efficiently working by using the concept of percentage of participation (POP) for mining frequent sequences in progressive databases with multiple minimum supports. Even though the algorithm is taking little more time for execution when compared with PISA (which finds the frequent sequential patterns in progressive databases by considering uniform min.support for different items) it works on frequent sequences with multiple minimum supports for different items. As the novel algorithm is finding the sequences with multiple minimum supports we can ignore the little increase in time of execution. The algorithm MS-PISA is very time efficient in that MS-PISA needs only one scan of the candidate structure mentioned by MMP-Tree at each time stamp rather quadratic scans. In addition with the novel design of MMP-Tree , MS-Pisa takes less space.

7. REFERENCES

[1] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," Proc. 20th Int'l Conf. Very Large Data Bases (VLDB '94), pp. 478-499, Sept. 1994.

[2] R. Agrawal and R. Srikant, "Mining Sequential Patterns," Proc. 11th Int'l Conf. Data Eng. (ICDE '95), pp. 3-14, Feb. 1995.

[3] R. Srikant and R. Agrawal, "Mining Sequential Patterns: Generalizations and Performance Improvements," Proc. Fifth Int'l Conf. Extending Database Technology (EDBT '96), Mar. 1996.

[4] B.Liu, W.Hsu and Y.Ma, "Mining association rules with multiple minimum supports", Proceedings of the fifth ACM, SIGKDD conference Sandiego,CA,USA,August 15-18,1999,P.341

[5] M.J. Zaki, "Efficient Enumeration of Frequent Sequences," Proc. Seventh ACM Int'l Conf. Information and Knowledge Management (CIKM '98), Nov. 1998.

[6] Jen-Wei Huang, Chi-Yao Tseng, Jian-Chih Ou, and Ming-Syan Chen, Fellow, IEEE: A General Model for Sequential Pattern Mining with a Progressive Database IEEE Transactions On Knowledge and Data Engineering VOL. 20, NO. 9, SEPTEMBER 2008

[7] S Cong, J. Han and D. Padua,"Parallel Mining of Closed Sequential Patterns," Proc. 11th ACM SIGKDD Int'l Conf. Knowl- edge Discovery and Data Mining (KDD '05), pp. 562-567, Aug. 2005.

[8] J. Wang and J. Han, "Bide: Efficient Mining of Frequent Closed Sequences," Proc. 20th Int'l Conf. Data Eng. (ICDE '04), pp. 79-91, 2004.

[9] X. Yan, J. Han, and R. Afshar, "Clospan: Mining Closed Sequential Patterns in Large Datasets," Proc. Third SIAM Int'l Conf. Data Mining (SDM '03), pp. 166-177, May 2003.

[10] M.N. Garofalakis, R. Rastogi, and K. Shim, "Spirit: Sequential Pattern Mining with Regular Expression Constraints," Proc. 25th Int'l Conf. Very Large Data Bases (VLDB '99), pp. 223-234, 1999.

[11] C. Luo and S.M. Chung, "Efficient Mining of Maximal Sequential Patterns Using Multiple Samples," Proc. Fifth SIAM Int'l Conf. Data Mining (SDM), 2005.

[12] H.Cao, N.Mamoulis, and D.W. Cheung, "Mining Frequent Spatio-Temporal Sequential Patterns," Proc. Fifth Int'l Conf. Data Mining (ICDM '05), pp. 82-89, Nov. 2005.

[13] J.-K. Guo, B.-J. Ruan, and Y.-Y. Zhu, "A Top-Down Algorithm for Web Log Sequential Pattern Mining," Proc. Ninth Pacific-Asia Conf. Knowledge Discovery and Data Mining (PAKDD), 2005.

[14] C.-C. Ho, H.-F. Li, F.-F. Kuo, and S.-Y. Lee, "Incremental Mining of Sequential Patterns over a Stream Sliding Window," Proc. IEEE Int'l Workshop Mining Evolving and Streaming Data IWMESD '06), Dec. 2006