International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-9 Issue-1, October, 2019

Pre-Order Post-Order Coded Aggregate Tree based Algorithm for Mining Sequential Patterns

Vemulapalli Saritha, Mogalla Shashi

Abstract*: *Sequential pattern mining is a data mining approach; aims to discover common interesting patterns in sequence datasets, which attracted a significant research interest due to its real world applications in various fields such as web click stream mining, retail business, stock market and bio-informatics. Each sequence in sequence dataset is composed of time ordered events and each event is an item set. It discovers all frequent subsequences having frequency greater than the given minimum support threshold. Discovering sequential patterns is expensive with respect to mining time as well as the amount of memory used, because of aggressive search space growth due to generation of explosive number of frequent subsequences with the sequence length as well as count of distinct items and large volume of sequence dataset. So, research in this domain aims at developing effective data structures which address frequency counting and large search space as well as scalable algorithms to reduce the execution time and the amount of memory utilized. We propose two efficient data structures called Pre-order Post-order Coded Aggregate Tree (PPCA-Tree) for compact representation of the sequence dataset and Root-node List of First-Occurrence Sub Trees Map (RLFOST-Map) for efficient representation of projected databases. We also developed an efficient Partially ordered Sequential PAttern Mining algorithm called PSPAM and Parallel implementation of Partially ordered Sequential PAttern Mining algorithm called PAPSPAM based on PPCA-Tree using RLFOST-Map which eliminates reconstruction of the projected databases. Experimental analysis done on various synthetic datasets proves that our algorithms PSPAM and PAPSPAM outperform prefixspan and other conventional & state-of-the-art algorithms over dense datasets with better scalability.

Keywords* : *Data mining, Pattern-growth, Sequential patterns, WAP-Tree.

I. INTRODUCTION

Sequential pattern mining (SPM) aims to discover frequently occurring time ordered events, where each event is an item set. Sequential patterns represent the interdependencies between items of the constituent events. SPM was initially introduced in [1], which attracted a significant research interest due to its real world applications in various fields. For example applying sequential pattern mining for post operative care and diagnosis of disease from the sequence of symptoms experienced (patient medical records), discovering customer buying patterns from customer shopping transaction sequences of retail store database and discovering user navigational patterns from web usage data for better website design and management.

Revised Manuscript Received on October 30, 2019.
* Correspondence Author

Vemulapalli Saritha*, Department of Information Technology, VNR Vignana Hyderabad
Dr. Mogalla Shashi, Department Of Computer Science And Systems Engineering, Andhra

A. *Sequential Pattern Mining Concepts*

Let a sequence dataset SD = $\{S_1,S_2,\dots ,S_i,\dots S_n\}$ where $S_1,S_2, ,\dots S_n$ are sequences and I = $\{i_1,i_2,\dots,i_m\}$ be an item set, where each sequence 'S_i' is an ordered events list denoted as $< (e_1)(e_2)\dots(e_i)..(e_k)>$ and each event 'e_i' is a subset of 'I'. The items in each event of a sequence in SD are arranged in lexicographic order, hence the occurrence of an item in an event is at most once, but occurrence can be many times in distinct events. The sequence 's' with length 'l' is called as l-length sequence. Length of sequence is defined as l=|s|, where $|s|= \Sigma^{k}_{j=1}|e_j|$. Let X=$< x_1\, x_2 \dots x_n >$ and Y=$< y_1\, y_2 \dots y_m >$ are the sequences, then 'X' is a subsequence of 'Y' and 'Y' contains 'X' denoted as $X\subseteq Y$, if $\exists n\in Z$ $1\leq j_1< j_2 <\dots< j_n \leq m$ such that $x_1 \subseteq y_{j1}$, $x2 \subseteq y_{j2}$, …, and $x_n \subseteq y_{jn}$. The support of 'X' is, the count of sequences in 'SD' which contains 'X'. Given a minimum support threshold 'min_sup', if the support of 'X' is greater than or equivalent to 'min_sup' then 'X' is a sequential pattern. In some of the applications of SPM such as bio-informatics in DNA sequences [2], in protein sequences [3] and web usage mining in web access data [4], where each event of a sequence is having single item is referred as totally ordered sequential pattern mining. General sequential pattern mining also referred as partially ordered sequential pattern mining, which is more complex problem. Algorithms for mining partially ordered sequential patterns are also able to mine totally ordered sequential patterns.

All SPM algorithms explore the patterns by performing s-extension and i-extension operations. These operations are used to generate candidate (k + 1)-length patterns from frequent k-length patterns. Let X=$< x_1\, x_2 \dots x_k >$ and Y=$< y_1\, y_2 \dots y_l >$ are the sequences, where k < l then X is a prefix of Y, if i) $x_i = y_i$ for 1 <= i <= (k-1) and ii) $x_k \subseteq y_l$ and x_k = first $|x_k|$ items of y_l. For instance (bc)(ab) is the prefix of (bc)(abc) whereas (b)(abc) is not. Let a sequence $s_p = < x_1\, x_2 \dots x_k >$, then s_{se} is stated as s-extension of the sequence s_p with singleton event z if $s_{se}=< x_1\, x_2 \dots x_k\, z>$ i.e s_p is a prefix of s_{se} and item z is added to a new event after last event of s_p . A sequence s_{ie} is stated as i-extension of the sequence s_p with item z if $s_{ie}=< x_1\, x_2 \dots (x_k \cup z)>$ i.e s_p is a prefix of s_{ie} and item "z" is added to the last event of s_p. For example (bc)(a) is an s-extension of (bc) with item "a" and (bc)(ab) is an i-extension of (bc)(a) with item "b".

B. *Sequential Pattern Mining Approaches*

All SPM algorithms prune the search space by applying apriori property, which states that all supersequences of an infrequent sequence cannot be frequent and hence pruned.

SPM algorithms are broadly classified as either Apriori based or Pattern-growth based algorithms. Apriori based algorithms explore the patterns using generate candidate-and-test strategy and adopts breadth-first search approach.

It first finds all frequent 1-length sequences then extends them with a frequent item to generate all frequent 2-length sequences by performing s-extension and i-extension operations and then generates all frequent 3- length sequences by extending frequent 2-length sequences with frequent item and so on until no more frequent sequences are generated. Pattern–growth based algorithms were found to be more promising due to limitations of apriori based algorithms towards scalability as they generate exponentially large number of candidates with increased length of sequences and number of distinct items and also because they require multiple scans of the database for finding support count of generated candidate sequence. Apriori based algorithms require more processing time and I/O cost due to generation of candidate patterns that does not occur in database.

Pattern-growth based algorithms employ divide-and-conquer paradigm for partitioning search space by introducing the concept of projected database and thereby reducing the database scan cost. Pattern–growth based algorithms adopt depth-first search approach. It first finds all frequent 1-length sequences then explore larger patterns by recursively building and scanning projected databases for each pattern to find its extensions by performing s-extension and i-extension operations with locally frequent items until the pattern can no longer be extended.

Conditional searching principal [5] is the basis for pattern-growth based algorithms. Let 'I' be an item set. Given an item 'p' ∈ I and sequence 's' that contains subsequence 'p' (i.e p ⊆ s), p-prefix of 's' is defined as the prefix of 's' from the beginning item to the first occurrence of 'p' and the remaining part after removing p-prefix is p-projection of 's'. For example a-prefix of (ca)(b)(abd) is (ca), the a-projection of (ca)(b)(abd) is (b)(abd) and d-prefix of (ca)(b)(abd) is (ca)(b)(abd) , the d-projection of (ca)(b)(abd) is €. Let 'SD' is a sequence dataset and 'p' is an item from I, the p-projected database of SD denoted as SD_p is the collection of p-projections of the sequences in 'SD' that contains 'p'.

Web Access Pattern Tree (WAP–Tree) based algorithms use WAP-Tree [6] to represent the sequence dataset and employ pattern-growth approach. These algorithms have shown remarkable performance in mining totally ordered sequential patterns from sequence dataset [7]. Since our proposed algorithm is based on WAP-Tree data structure and pattern-growth approach, some existing WAP–Tree based algorithms are explored in subsequent section.

C. *WAP-Tree based Approaches*

WAP-Tree [6] data structure is developed for representing the totally ordered sequence dataset. Nodes of WAP-Tree, except the root node which is empty, represent only frequent items. Each sequence in the dataset is inserted into the WAP-Tree starting from root node by incrementing count for shared prefix sequence nodes and adding nodes for unshared suffix sequence. Each node representing a frequent item maintains a count of the sequences sharing the common path from root to the node, a reference to its first child node, and a reference to its immediate next right sibling node.

First-Occurrence Forest (FOF) [8] data structure is designed to represent the projected database and for finding the support count of a prefix-pattern efficiently in WAP-tree based algorithms. Given an item 'i', first-occurrence subtrees of 'i' is defined as the set of subtrees rooted at first-occurrence of 'i'. A node in WAP-tree is called as a first-occurrence of item 'i', if it is marked with label 'i' and no one of its ancestor nodes marked with label 'i'. FOF represents first-occurrence subtrees forest, which is list of pointers to the root nodes of the first-occurrence subtrees. The subtrees rooted at child nodes of first-occurrence of prefix 'p' represent p-projected database. This approach [8] first find all frequent 1-length patterns by reading the database once and then constructs WAP-Tree which represents the sequence dataset. For each frequent pattern it explores larger patterns by appending with locally frequent items appearing in its projected database by recursively searching WAP-Tree starting from nodes in FOF of current pattern(s) using pattern-growth strategy. In Root-set of Suffix Trees (RST) based approach [9] the authors proposed the data structure called RST to find web access patterns from web log database. This approach first find all frequent 1-length patterns by reading the database once and then constructs WAP-Tree which represents the sequence dataset. For finding frequent patterns this approach accumulates first occurrences of frequent items found during traversing the WAP-Tree using depth-first search strategy by maintaining a table with all frequent items and its root-sets which contain first occurrence nodes information. For each prefix-pattern, it explores larger patterns by appending with frequent 1-length patterns appearing in its projected database by recursively exploring the WAP-Tree starting from its root-set nodes using pattern-growth strategy. This approach is more efficient; as it needs less numbers of tree traverses compared to the other WAP-Tree based algorithms. All the frequent patterns with same prefix-pattern are generated in a single traversal of the WAP-tree, whereas FOF approach [8] explores the WAP-tree for each prefix-pattern.

MULTI-FOF-SP [10] is MULTI-WAP-Tree [10] based algorithm, inspired by FOF and integrates pruning approach called "Sibling Principle". MULTI-WAP-Tree [10] data structure is an extension of WAP-Tree, which represent partially ordered sequence dataset. Unlike WAP-tree the nodes of MULTI-WAP-Tree are connected with two different types of edges to represent s-extension and i-extension of a prefix sequence. In addition to the data accommodated in WAP-Tree node, MULTI-WAP-Tree node also contains a link to the parent node. This approach finds all frequent 1-length sequences and then constructs MULTI-WAP-Tree. This approach finds projected databases using first-occurrence forest. For each frequent pattern it explores larger patterns by performing s-extension and i-extension operations with frequent 1-length patterns appearing in its projected database by recursively searching MULTI -WAP-Tree using depth-first search strategy starting from nodes in FOF of current pattern(s) using pattern-growth strategy. In this approach separate tree traversal is required for growing a prefix-pattern with each possible extension. In this paper the authors propose an efficient algorithm which can extract partially ordered sequential patterns without requiring repeated tree traversals for different possible extensions of a prefix-pattern combining the merits of Root-set of Suffix Trees (RST) based mining approach [9] and MULTI-FOF-SP [10] based algorithms.

For this purpose the authors devised two new data structures namely Pre-order Post-order Coded Aggregate Tree (PPCA-Tree) and Root-node List of First Occurrence Sub-Trees Map (RLFOST-Map) for compact representation of sequence database and for representing projected database respectively.

The rest of the paper is organized into four sections. Literature review & recent advancements are presented in Section 2. The proposed algorithms PSPAM and PAPSPAM using the new data structures Pre-order Post-order Coded Aggregate Tree (PPCA-Tree) and Root-node List of First Occurrence Sub-Trees Map (RLFOST-Map) are introduced in Section 3. The experimental results in support of the effectiveness & efficiency of our proposed algorithms are discussed in section 4. Finally, conclusions in section 5.

II. Literature Review

Overviews of methodologies employed by most popular algorithms are discussed below.

CM-SPADE [11] and CM-SPAM [11] decrease the count of candidate patterns and join operations using Co-occurrence pruning [11], which prune the candidates using information in Co-occurrence Map (CMAP) structure. CMAP structure associates each frequent item to its list of succeeding items by i-extension and s-extension operations. CM-SPADE and CM-SPAM outperforms GSP, Spam, BitSpade and prefixspan. CM-Spade is claimed as current fastest algorithm [11],[12].

Prefixspan [13] is the most successful and widely used pattern-growth based algorithm. Prefixspan adopts depth-first search approach. It first finds all frequent 1- length sequences. It explores larger patterns by recursively constructing and scanning projected databases of current pattern(s) until no frequent sequences are generated. It explores larger patterns by recursively doing s-extension and i-extension operations with frequent 1-length sequences appearing in the projected database in lexicographical order. It reduces the cost required for reconstructing the projected databases by introducing the concept of pseudo-projection, which is implemented as set of pointers to sequential dataset. Prefixspan algorithm mine sequential patterns from large sequence datasets and is used as bench mark for comparing the performance of proposed algorithms.

SPMF [14] Library provides benchmark implementations of various SPM algorithms. Efficient variant of prefixspan algorithm is implemented by Philippe Fournier-Viger, et al. and made it available in SPMF library which is taken as base line for comparing the proposed algorithms.

Several other variant SPM algorithms such as Closed sequential pattern mining algorithms [11],[15],[16] and Maximal sequential pattern mining algorithms [17],[18] are proposed to find set of meaningful patterns which are compact representations of complete set of patterns.

Most of the algorithms consider all items are of equal importance and frequency as a measure for finding sequential patterns. In some real world applications, a sequence datasets usually grow over time and also uses utility, risk, profit, weight, etc as a measure to find the importance of items. An efficient algorithms IncUSP-Miner^{+} [19], AHUS&AHUS-P [20] are developed for finding high utility sequential patterns by considering frequency and utility measures.

III. Proposed Methodology

Developing efficient data structures and scalable algorithms to reduce mining time and amount of memory required is a research issue in sequential pattern mining due to exponential search space and large volume of sequence datasets. The proposed framework addresses both the challenges of pattern-growth algorithms namely support counting and intermediate results maintenance simultaneously by developing efficient data structures that play an essential role in improving scalability of sequential pattern mining.

The authors propose an efficient WAP-Tree based algorithms, Partially ordered Sequential PAttern Mining (PSPAM) and Parallel implementation of Partially ordered Sequential PAttern Mining (PAPSPAM) by taking the advantages of multi-core processor architecture, for mining partially ordered sequential patterns from dense datasets. It borrows concepts from existing algorithms RST based mining approach [09] and MULTI-FOF-SP [10] which are discussed in section 1.

Data structure Pre-order Post-order Coded Aggregate Tree (PPCA-Tree) is devised to accommodate the essential information for mining the sequential dataset. It represents the sequence dataset in a compact manner for deriving efficiency in terms of memory space and CPU time. It is a trie like structure where in each sequence is represented along a shared path so that order is maintained while being compact. The post-order and pre-order sequence numbers of nodes in a tree are used to find ancestor-descendant relationship between the nodes for candidate pruning during the mining process. Another data structure RLFOST-Map is also devised to represent projected databases by maintaining a map with all frequent items and its root-node lists which contain the list of pointers to the root nodes of first-occurrence subtrees.

A. *An abstract view of proposed approach*

The following are the fundamental steps for mining partially ordered sequential patterns using PPCA-Tree and RLFOST-Map.

1. Find all frequent items “Σ” from the given input sequence dataset.
2. Construct pre-order post-order coded aggregate tree, which represent the subsequences formed by removing all infrequent items from the input sequence dataset.
3. Mine partially ordered sequential patterns from pre-order post-order coded aggregate tree using pre-order, post-order coding of nodes and RLFOST-Map using the algorithm proposed by authors which adapts the ideas from existing algorithms RST based mining approach [09] and MULTI-FOF-SP [10].

B. *PPCA-Tree: Design and Construction*

An efficient data structure called pre-order post-order coded aggregate tree (PPCA-Tree) is proposed in this section, which is an extension of MULTI-WAP-Tree [10] is proposed. PPCA-Tree accommodates the sequence dataset in compressed format.

➢ *Pre-order Post-order coded aggregate tree is described as below.*

1. Each node represents an item and accommodates information in the following fields.
   i) label: contains an item name.

ii) count: contains the no. of sequences which share a common path from root to the node of a tree.
iii) eventno: contains the event number to which the item represented by the node belongs.
iv) precode: contains pre-order number representing position of the node in the pre-order sequence of the nodes of a tree
v) postcode: contains post-order number representing position of the node in the post-order sequence of the nodes of a tree
vi) leftchild: holds a reference to its first child node.
vii) rightsibling: holds a reference to its immediate next right sibling node.

2. The nodes of PPCA-Tree are connected with two different types of edges SE-Edge and IE-Edge to represent s-extension and i-extension respectively of a prefix sequence.
3. Given any two nodes N1 and N2; N2 is descendant of N1 if N1.precode < N2.precode and N1.postcode > N2.postcode then N1 & N2 exists in the same branch of the tree.
4. Given any two nodes N1 and N2, if N1.precode < N2.precode and N1.postcode < N2.postcode then it is implied that N1 & N2 exists in different branches of the tree and hence ancestor-descendant relationship does not exist between N1 & N2.

➢ *Steps for constructing Pre-order Post-order coded aggregate tree*

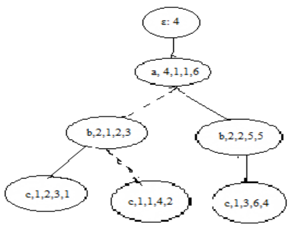
1. Create root node with label as €, set its eventno and count as 0.
2. for each sequence in sequence dataset
   i) prune the sequence by removing all infrequent items
   ii) insert the pruned subsequence in the pre-order post-order coded aggregate tree starting from root node by incrementing the count for shared prefix sequence nodes (by checking the equality of both node label and edge category) and inserting new nodes for unshared suffix sequence.
3. Assign pre-order code for the nodes of a tree by traversing the tree in pre-order.
4. Assign post-order code for the nodes of a tree by traversing the tree in post-order.

Fig. 1 shows the PPCA-Tree for the sequence dataset shown in Table 1 with the min_sup=0.5%.

Table 1: Sample Sequence Dataset 1

sid	sequence	Frequent subsequene
1	(abd)(c)	(ab)(c)
2	(a)(b)(c)	(a)(b)(c)
3	(af)(b)	(a)(b)
4	(abc)(e)	(abc)

Fig. 1: PPCA-Tree for Dataset in Table



C. *Mining Partially Ordered Sequential Patterns based on PPCA-Tree and RLFOST-Map*

Proposed PSPAM and PAPSPAM Algorithms mine partially ordered sequential patterns from PPCA-Tree using pre-order, post-order coding of nodes. An efficient data structure called Root-node List of First occurrence Sub Trees Map (RLFOST-Map) is proposed for representing projected databases in order to avoid repeated traversals of the projected database for various possible extensions of a growing pattern. The proposed pattern-growth based algorithms PSPAM and PAPSPAM uses the new data structures PPCA-Tree and RLFOST-Map for achieving scalability and competitive performance while finding frequent sequential patterns.

The PPCA-Tree contains two types of edges: SE_Edge and IE_Edge representing s-extension and i-extension respectively. With the existence of 2 different types of edges SE-Edge & IE-Edge in PPCA-tree require distinction between s-occurrence and i-occurrence of an item. s-occurrence form sequence extension and i-occurrence form item set extension. The occurrence type of a node, in a path from the given root node is determined based on the following rules [10].

i. If at least one SE-Edge exists, then occurrence type is s-occurrence.
ii. If only IE-Edges exist, then occurrence type is i-occurrence.
iii. If the last item set of prefix-pattern is identified in its ancestor nodes before an occurrence of SE-Edge, then occurrence type is i-occurrence.

➢ *Root-node List of First-occurrence SubTrees Map (RLFOST-Map)*

RLFOST-Map represent projected databases, by maintaining a map with all frequent items and their root-node lists which holds list of pointers to the root nodes of subtrees rooted at first-occurrences of a prefix-pattern (projected database). RLFOST-Map eliminates reconstruction of aggregate trees represent projected databases for each prefix-pattern, which helps in reducing the memory utilization. Given a PPCA-Tree and frequent item 'p' for a sequence dataset SD, the p-projected database of SD denoted as SD_p is represented by the subtrees rooted at first-occurrences of a prefix-pattern 'p'. A node is said to be first-occurrence if none of its ancestor nodes has the same label in the projected database.

Proposed pattern-growth based approach finds frequent sequential patterns by traversing the subtrees rooted at first-occurrences for each prefix-pattern in the depth-first search order. Table 2 shows RLFOST-Map for the prefix-pattern 'a'.

Table 2: RLFOST-Map for the prefix-pattern 'a'

b	**(b,2,1,2,3), (b,2,2,5,5)**
c	**(c,1,2,3,1),(c,1,1,4,2),(c,1,3,6,4)**

➢ *Proposed PSPAM and PAPSPAM Algorithms*

PSPAM and PAPSPAM algorithms follows divide-and-conquer approach, recursively divides the search space into multiple sub-search spaces and extracts frequent patterns from each individual sub-search space. Finally, these individual frequent patterns are combined to obtain the final set of frequent patterns.

PAPSPAM algorithm extracts frequent patterns from each individual sub-search space by executing simultaneously on different cores of the processing device.

a) The basic steps for extending frequent sequential pattern 'α' ∈ Σ

1. Scan PPCA-Tree once to find ∀p ∈ Σ p-projected database of SD denoted as SD_p containing list of pointers to the root nodes of subtrees rooted at first-occurrences of a prefix-pattern α of PPCA- Tree.
2. Scan p-projected database SD_p once, to find
   i) Frequent items set 's-ext-set' so that an item i in 's-ext-set' can be appended to 'p' to form a s-extension of 'p'.
   ii) Frequent items set 'i-ext-set' so that an item 'i' in 'i-ext-set' can be appended to last event of 'p' to form a i-extension of 'p'.
3. For each item 'i' in 's-ext-set'
   Build sequential pattern 'p' by appending 'i' to 'p'.
   Build p–projected database SDp and recursively execute the steps from 2 to 4.
4. For each item 'i' in 'i-ext-set'
   Build sequential pattern 'p' by appending 'i' to 'p'.
   Build p–projected database SDp and recursively execute the steps from 2 to 4.

b) Proposed pruning strategies

Our proposed PSPAM & PAPSPAM algorithms prune the candidate item set of a grown-pattern using sibling principle, in which apriori principle is interpreted using lexicographic pattern tree. Let N is a node represents a grown-pattern in the lexicographic pattern tree & S is its sibling node, sibling principle states that if S is not frequent; candidate sequential patterns formed by extending the grown pattern represented by N with S is also infrequent. Proposed PSPAM and PAPSPAM algorithms extends sibling principle based pruning strategy for pruning candidate items using ancestor relationship and also by finding the estimated support count based on the concept of checkpoint. These pruning strategies improve the performance by reducing the search space. For any nodes N1 and N2 existing in the same path of the tree; N2 is descendant of N1 if N1.precode < N2.precode and N1.postcode > N2.postcode. Proposed pruning strategies prune the candidate items using ancestor relationship and also by finding the estimated support count for all candidate items based on checkpoint value while searching for its first occurrence nodes by traversing PPCA-tree in pre-order traversal. It searches for first occurrence nodes by processing root node first then recursively processing the subtree rooted at its first child until either all nodes are visited or first occurrence of all candidate items are identified and then prune candidate items using ancestor relationship before processing any subtree rooted at its other children.

Checkpoint & estimated support count are defined as:
Checkpoint = |Projected Database| - min_sup + 1.
Estimated Support count (est_sup) = support of candidate item at checkpoint+|Projected Database| - checkpoint.
Proposed algorithms prune the candidate items whose estimated support count (est_sup) value is less than min_sup. Because all new candidate items occur in projected database after checkpoint cannot be frequent.

IV. EXPERIMENTATION AND RESULTS

Experimental analysis was done on various synthetic datasets under varying minimum support thresholds inorder to assess the efficiency of proposed algorithms PSPAM and PAPSPAM with Prefixspan and other conventional & state-of-the-art algorithms. Experimental analysis was conducted on Intel(R) Xeon(R) CPU E3-1225V5@3.30-GHz,32GB RAM Quad- core processor running on Microsoft Windows 10. Execution time of proposed algorithms is compared with implementation of Prefixspan and other conventional & state-of-the-art algorithms (SPAM, SPADE, CM-SPAM, CM-SPADE) provided in SPMF, which are implemented in java. Synthetic dataset generator in SPMF is used to generate various synthetic datasets with various parameters. Various parameters represent : N is the count of unique items, D is the count of sequences, T is the count of items in an event and C is the count of events in a sequence to be generated. Synthetic datasets are named correspondence to various parameter values. For example C3T3N5D100K specifies a dataset generated with various parameter values C=3, T=3, N=5 and D=100K (in thousands). Table 3 to 4 shows execution time (in milliseconds) of various algorithms on synthetic datasets C3T3N5D100K, C4T3N5D100K under varying minimum support thresholds respectively.

Table

Sup port	Prefix span	SP AM	PS PAM	PAP SPAM	SPA DE	CM-SPADE	CM-SPAM
0.5	902	1832	363	410	430	601	2809
0.4	924	1900	387	425	830	930	2838
0.3	1113	2940	393	428	874	1003	4670
0.2	1907	10040	415	444	1855	1630	6376
0.1	2932	18228	469	525	2626	3419	15869
0.05	3413	22907	524	597	5530	5338	18374
0.025	3863	23678	570	642	7283	6973	18483
0.01	4034	22100	586	662	9839	9210	17637

Table 4

Sup port	Prefix span	SPAM	P SPAM	PAP SPAM	SPA DE	CM-SPADE	CM-SPAM
0.5	2169	6063	1026	1079	1065	1365	4339
0.4	3069	10347	1350	1294	2195	2420	12572
0.3	3833	13868	1574	1389	4076	3274	14641
0.2	7312	28742	2608	2358	4346	4227	29283
0.1	12382	34802	3959	2485	10312	9945	37053
0.05	19521	75407	5155	3380	20339	18885	75235
0.025	23799	107392	6856	4738	38123	38429	99716
0.01	28952	147057	8492	5203	84628	84899	141783

Fig. 2 to 3 compare the execution time performance of proposed PSPAM and PAPSPAM algorithms with Prefixspan and other conventional & state-of-the-art algorithms (SPAM, SPADE, CM-SPAM, CM-SPADE) on synthetic datasets C3T3N5D100K, C4T3N5D100K under varying minimum support thresholds respectively. Each figure, x-axis indicates minimum support threshold values in percentage and y-axis indicate execution time in milliseconds.

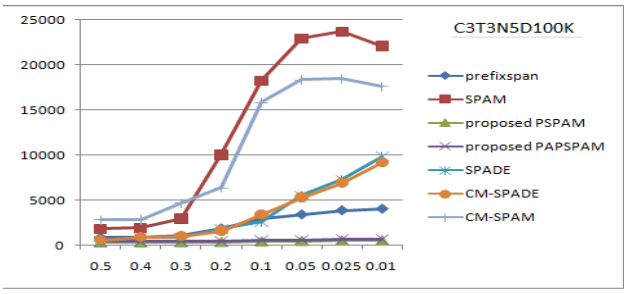


Fig 2.

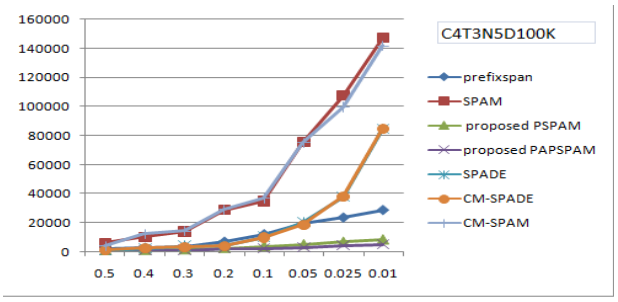


Fig. 3.

In [24] the authors proved that CM-Spade & CM-Spam algorithms outperform state-of-the-art algorithms (GSP, Prefixspan, SPADE, SPAM) on various totally ordered real-life datasets. Based on the above observations we have identified prefixspan, CM-Spade & CM-Spam are closest competitors to the proposed algorithms. Table 5 to 6 shows execution time (in milliseconds) of various algorithms on synthetic datasets C5T3N5D300K, C3T4N5D100K under varying minimum support thresholds respectively. Fig. 4 to 5 compare the execution time performance of proposed algorithms PSPAM and PAPSPAM with Prefixspan, CM-Spade & CM-Spam on synthetic datasets C5T3N5D300K, C3T4N5D100K under varying minimum support thresholds respectively. The candidate patterns count increases with sequence length as well as count of distinct items. For a fixed size sequential dataset as the candidate patterns count increases the density reduces and hence becomes sparser. By extending sequential dataset with the additional sequences its density can be maintained. The results prove that our proposed PSPAM and PAPSPAM algorithms outperform Prefixspan and other conventional & state-of-the-art algorithms (SPAM, SPADE, CM-SPAM, CM-SPADE) with respect to execution time over dense datasets with better scalability.

Table 5

Supp ort	Prefix span	CM-SPADE	CM-SPAM	PSPAM	PAP SPAM
0.5	16566	14377	39809	12864	8415
0.4	38755	18038	95536	22388	11751
0.3	50279	37110	150704	36408	19356
0.2	74202	68694	191726	51289	24783
0.1	230006	172416		130155	54075
0.05	315506	298736		192074	98898
0.025	440503	525052		283052	133512
0.01	610052	1343873		394327	223898

Table 6

Supp ort	Prefix span	CM-SPADE	CM-SPAM	PSPAM	PAP SPAM
0.5	5895	4206	13690	415	403
0.4	6641	3510	17933	425	414
0.3	9333	7158	32863	433	442
0.2	16358	15036	54350	501	527
0.1	24534	25948	86246	586	599
0.05	28179	33153	91157	620	639
0.025	28678	36321	92489	655	652

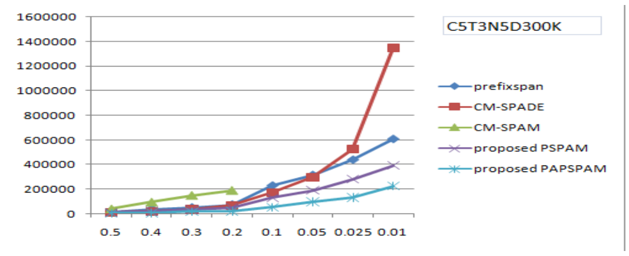


Fig. 4.

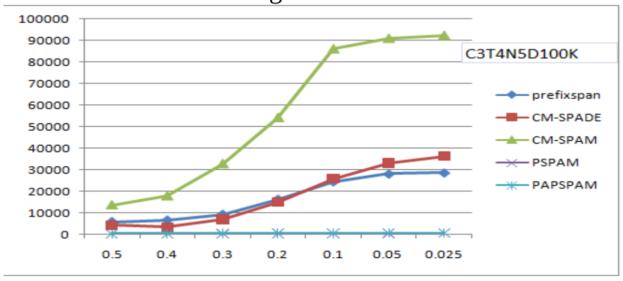


Fig. 5.

V. CONCLUSIONs

Sequential pattern mining contributes by providing solutions for numerous real time applications in various domains such as web click stream mining, retail business, stock market, e-commerce, bio-informatics, e-learning and telecommunications. The proposed framework addresses both the challenges of pattern-growth algorithms namely support counting and intermediate results maintenance simultaneously by developing efficient data structures that play an essential role in improving scalability of sequential pattern mining. we proposed two efficient data structures called Pre-order Post-order Coded Aggregate Tree (PPCA-Tree) for compact representation of sequence dataset and Root-node List of First Occurrence Sub Trees Map (RLFOST-Map) for representing the projected databases in order to avoid repeated traversals of the projected database for various possible extensions of a growing pattern. We also proposed an efficient partially ordered sequential pattern mining algorithms called PSPAM and parallel implementation of partially ordered sequential pattern mining algorithm called PAPSPAM (by taking the advantages of multi-core processor architecture) based on PPCA-Tree using RLFOST-Map.

Proposed pruning strategies prune candidate items using ancestor relationship and also by finding the estimated support count based on the concept of checkpoint inorder to increase the efficiency by condensing the search space. The candidate patterns count increases with sequence length as well as count of distinct items. For a fixed size sequential dataset as the candidate patterns count increases the density reduces and hence becomes sparser. By extending sequential dataset with the additional sequences its density can be maintained. Experimental analysis done on various synthetic datasets under varying minimum support thresholds, results proves that our proposed algorithms PSPAM and PAPSPAM outperforms Prefixspan and other conventional & state-of-the-art algorithms (SPAM, SPADE, CM-SPAM, CM-SPADE) with respect to runtime on dense datasets with better scalability.

REFERENCES

1. R. Agrawal and R. Srikant, “Mining sequential patterns”, In Proceedings of 11th International Conference on Data Engineering, pp. 3–14, IEEE ,1995.
2. K. Wang, Y. Xu, and J.X. Yu, ”Scalable sequential pattern mining for biological sequences”, In Proceedings of the 13th ACM international conference on Information and knowledge management, pp. 178–187, ACM, 2004.
3. T. P. Exarchos, C. Papaloukas, C. Lampros, and D.I. Fotiadis, “Mining sequential patterns for protein fold recognition”, Journal of Biomedical Informatics, pp.165–179, 2008.
4. M.K. Khribi, M. Jemni, and O. Nasraoui, ”Automatic recommendations for e-learning Personalization based on web usage mining techniques and information retrieval”, In 8th IEEE International Conference on Advanced Learning Technologies (ICALT’08), pp. 241–245. IEEE, 2008.
5. P. Tang, M.P. Turkia, and K.A. Gallivan, “Mining web access patterns with First- Occurrence linked WAP-trees”, In Proceedings of the 16th International Conference on
6. Software Engineering and Data Engineering (SEDE’07), pp. 247–252, 2007.
7. J. Pei, J. Han, B. Mortazavi-Asl, and H. Zhu, “Mining Access patterns efficiently from web Logs”, Knowledge Discovery and Data Mining. Current Issues and New Applications, pp. 396–407, 2000.
8. N.R. Mabroukeh and C.I. Ezeife, “A taxonomy of sequential pattern mining algorithms”, ACM Computing surveys (CSUR), vol. 43(1), pp. 3:1-3:41, 2010.
9. Peterson E and Tang P, “Mining frequent sequential patterns with first-occurrence forests”, In Proceedings of the 46th Annual Southeast Regional Conference (ACMS-46E), pp. 34–39, ACM, 2008.
10. Manira Akhter, Ashin Ara Bithi and Abu Ahmed Ferdaus, ”Mining Web Access Patterns using Root-set of Suffix Trees”, International Journal of Computer Applications, Vol. 94(9), pp. 23-29, 2014.
11. K. D. Onal and P. Karagoz, “Extracting Multi-item Sequential Patterns by Wap-tree Based Approach”, WEBIST- International Conference on Web Information Systems and Technologies, pp. 215-222, 2014.
12. P. Fournier-Viger, A. Gomariz, M. Campos, and R. Thomas, “Fast Vertical Mining of Sequential Patterns Using Co-occurrence Information”, The Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), pp. 40-52, 2014.
13. P. Fournier-Viger, Jerry Chun-Wei Lin, R.U. Kiran, Y.S. Koh, “A Survey of Sequential Pattern Mining”, DataScience and Pattern Recognition, pp. 54-77, Vol. 1(1), 2017.
14. J. Pei, J. Han, B. Mortazavi-Asl, J. Wang, H. Pinto, Q. Chen, U. Dayal, and M. C. Hsu, “Mining sequential patterns by pattern-growth: The prefixspan approach", IEEE Transactions on knowledge and data engineering, vol. 16(10), pp. 1424-1440, 2004.
15. P. Fournier-Viger, C. W. Lin, A. Gomariz, A. Soltani, Z. Deng, H. T. Lam, “The SPMF open-source data mining library version 2", The European Conference on Principles of Data Mining and Knowledge Discovery, pp. 36-40, 2016, URL: http://www.philippe-fournier-viger.com/spmf/.
16. A. Gomariz, M. Campos, R. Marin and B. Goethals, “ClaSP: An Efficient Algorithm for Mining Frequent Closed Sequences”, The Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD, Part I), LNCS, vol. 7818, pp. 50–61, 2013.
17. F. Fumarola, P. F. Lanotte, M. Ceci, and D. Malerba, “CloFAST: closed sequential pattern mining using sparse and vertical id-lists", Knowledge and Information Systems, vol. 48, pp. 429-463, 2016.
18. P. Fournier-Viger, Cheng-Wei Wu, A. Gomariz and V.S. Tseng, “VMSP: Efficient Vertical Mining of Maximal Sequential Patterns”, Canadian Conference on Artificial Intelligence, LNAI, vol. 8436, pp. 83–94, 2014.
19. P. Fournier-Viger, Cheng-Wei Wu and V.S. Tseng, “Mining Maximal Sequential Patterns without Candidate Maintenance”, In International conference on Advanced data mining and applications (ADMA) , LNCS, vol. 8346, pp. 169–180, 2013.
20. Jun-Zhe Wang and Jiun-Long Huang, ”IncUSP-Miner+: On Incremental high utility sequential pattern mining", ACM Transactions on Intelligent Systems and Technology (TIST), Vol. 9(55), 2018.
21. B. Le, U. Huynh, Duy-Tai Dinh, “A pure array structure and parallel strategy for high-utility sequential pattern mining”, Expert Systems With Applications, pp.107-120, 2018.

AUTHORS PROFILE

Vemulapalli Saritha received the M.Tech degree in Information Technology (2002) from Andhra University College of Engg, Andhra University, Visakhapatnam, Andhra Pradesh, India. She is currently pursuing Ph.D degree in Computer Science And Systems Engineering from Andhra University College of Engg, Andhra University. She is currently working as Asst. Prof. in Department of Information Technology, VNR Vignana Jyothi Inst. Of Engg. & Tech., Hyderabad, Telangana, India. Her research interests include Data Mining, Sequential pattern mining and Web usage mining.

Dr. Mogalla Shashi received the Ph.D degree in Computer Science And Systems Engineering (1994) from Andhra University College of Engg, Andhra University, Visakhapatnam, Andhra Pradesh, India. She is currently working as professor in Department Of Computer Science And Systems Engineering, Andhra University College of Engg., Andhra University. She has published more than 100 research papers in international / national journals and conferences. Her research interests include Data Mining, Artificial Intelligence and Information Security.