An Integrated Approach for Mining Closed Sequential Patterns

V. Purushothama Raju

Research Scholar, Department of CSE
Acharya Nagarjuna University
Guntur, A.P., India

Dr. G.P. Saradhi Varma

Department of Information Technology
S.R.K.R. Engineering College
Bhimavaram, A.P., India

Abstract—Sequential pattern mining has been a well studied area in data mining for over a decade. Yet researchers are still uncovering interesting problems, new algorithms and ways to improve upon existing methods. To efficiently mine long sequential patterns, closed sequential pattern mining was introduced. In this paper, we propose an efficient closed sequential pattern mining algorithm CSPAM by integrating the best qualities of an efficient sequential pattern mining algorithm SPAM and an efficient closed itemset mining algorithm closet+.

Keywords-Data Mining, Sequential Pattern Mining, Closed Sequential Pattern Mining.

I. INTRODUCTION

Mining sequential patterns has attracted a significant amount of research. Popularity in this area is primarily due to its large area of applicability. Sequential pattern mining can be used for mining customer shopping sequences, biological sequences, web click streams, XML query access patterns for caching, block correlations in storage systems, API usages from open source software, sequences of file block references in operating systems and network intrusion detection. Other applications of sequential pattern mining include target marketing, customer retention, feature selection for sequence classification, user behavior analysis, finding copy-paste and related bugs in large software, study of engineering and medical process, personalization systems and web recommender systems.

Sequential pattern mining algorithms are inefficient at mining long sequences. Long sequences generate exponential number of sub sequences, for example a long frequent sequence $\{(x_1)(x_2)\ldots.(x_{50})\}$ will generate 2^{50}-1 subsequences. The performance of sequential pattern mining algorithms degrades when mining at low support values. Closed sequential pattern mining was proposed to overcome the limitations of sequential pattern mining algorithms. Closed sequential pattern mining produces more compact result set than sequential pattern mining and also offers better efficiency for mining long sequences. Only a few algorithms were proposed for mining closed sequential patterns, this is due to the complexity of the problem.

In this paper, we integrate an efficient sequential pattern mining algorithm SPAM[6] and an efficient closed itemset mining algorithm closet+[15] in an attempt to develop a new closed sequential pattern mining algorithm CSPAM with the best qualities of the aforementioned algorithms. Our algorithm CSPAM outperforms FMCSP[16] by an order of magnitude.

The rest of this paper is organized as follows. Section 2 discusses the related work. Section 3 presents the proposed method. Section 4 reports the performance evaluation. Finally, we conclude the work in Section 5.

II. RELATED WORK

Agrawal and Srikant [1] introduced the problem of sequential pattern mining. Later, efficient algorithms such as GSP [2], SPADE [5] and SPAM [6] were proposed based on Apriori approach. Apriori-based algorithms follow a candidate maintenance-and-test paradigm, which exploits the downward closure property. Other algorithms such as FreeSpan [4] and PrefixSpan [7] follow a pattern-growth based approach. Pattern-growth based algorithms use an incremental approach in generating possible frequent sequences and make projections of the database to reduce the search space.

Closed itemset mining was proposed to mine closed itemsets without any superset with the same support. Closed itemset mining can produce smaller result set than frequent itemset mining with the same expressive power. Closed itemset mining algorithms like CLOSET[12] and CHARM [13] adopt space efficient depth first search. CLOSET adopts a compressed database representation called FP-tree to mine closed itemsets. CHARM adopts a compact vertical tid list structure called diffset to mine closed itemsets.

CLOSET+[15] combines the merits of the previously developed effective strategies and new concepts the item skipping technique and efficient subset-checking scheme. Item skipping technique further prunes search space and speeds up mining. The subset-checking scheme saves memory usage and accelerates the closure-checking significantly. CLOSET+ performs better than CLOSET and CHARM in terms of scalability, memory utilization and execution time.

There are only two popular algorithms CloSpan [8] and BIDE[11] in closed sequential pattern mining. CloSpan produces a candidate set for closed sequential patterns and performs post pruning on it. CloSpan requires more storage to store the closed sequence candidates when mining long patterns or the support threshold is low and it offers poor scalability. BIDE adopts the framework of PrefixSpan and uses BackScan pruning method to stop growing redundant patterns. BIDE is a computational intensive approach since it requires more number of database scans for the bi-direction closure checking and the BackScan pruning.

III. Proposed Method

A. *Data Representation*

We use a vertical bitmap representation of the data for efficient counting of support. A vertical bitmap is constructed for each item in the dataset, and each bitmap has a bit corresponding to each itemset in the sequence. If item *k* occurs in itemset *p*, then the bit corresponding to itemset *p* of the bitmap for item *k* is set to *one.* Otherwise, the bit is set to *zero*. We divide the bitmap in such a way that all of the itemsets of each sequence in the database will present together in the bitmap as shown in Figure 1. If itemset *i* appears before itemset *j* in a sequence, then the index of the bit *i* is made smaller than that of the bit *j*.

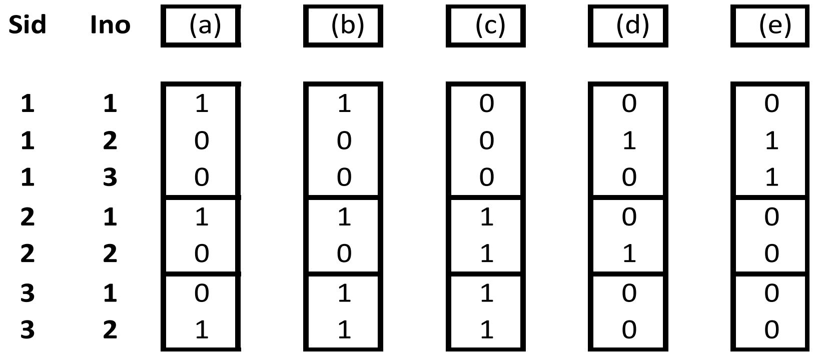
The bitmap for the itemset{i, j} is the bitwise *AND* of bitmap for item *i* and a bitmap for item *j*. If the last itemset of the sequence is in transaction *j* and all other itemsets of the sequence present in transactions before *j*, then the bit corresponds to *j* is set to *one.* Otherwise, it is set to *zero*.

We divide the customer sequences into different sets based on their lengths. If the length of a sequence is between $2^n + 1$ and 2^{n+1} then we treat it as a 2^{n+1} bit sequence. The minimum value of *n* is set to 1. Each set of 2^n bit sequences will represent a different bitmap, and in that bitmap each section will be 2^n bits long. Support counting is a simple check to determine whether the corresponding bitmap partition contains all zeros or not.

TABLE I. A Sample Sequence Database

Sid	Sequence
1	(ab)(de)(e)
2	(abc)(cd)
3	(bc)(abc)

The bitmap representation of the dataset that is shown in Table 1 is given in Fig. 1. Each section in the vertical bitmap represents a customer's sequence. The itemset 1 in sequence 2 contains items a,b and c, so the bit that corresponds to that itemset in each of the bitmaps a, b and c is set to one. Since itemset 1 does not contain the items d and e, the bits corresponding to that itemset in bitmaps d and e are set to zero.



Sid	Ino	(a)	(b)	(c)	(d)	(e)
1	1	1	1	0	0	0
1	2	0	0	0	1	1
1	3	0	0	0	0	1
2	1	1	1	1	0	0
2	2	0	0	1	1	0
3	1	0	1	1	0	0
3	2	1	1	1	0	0

Figure 1. Vertical bitmap representation of a sample sequence database

B. *Lexicographic Sequence Tree*

We use sequence lattice framework to explain our algorithm. The items in the database are arranged in the lexicographical order. If item *a* appears before item *b* in the ordering, then we represent this by $a \leq b$. if α is a subsequence of β then it is represented as $\alpha \leq \beta$. Assume all sequences in the database are arranged in a lexicographic sequence tree. The root of the tree is represented using ϕ. If *k* is a node in the tree then it's children are all nodes *k′* follow lexicographic ordering $k \leq k'$.

There are two types of sequence extensions namely sequence-extended sequence and itemset-extended sequence. A sequence-extended sequence is produced by adding an itemset containing a single item to the end of its parent's sequence. An itemset-extended sequence is produced by adding an item to the last itemset in the parent's sequence. For example, if α = {(abc),(cd)}then {(abc),(cd),(e)}is a sequence-extended sequence of α and {(abc),(cde)}is an itemset-extended sequence of α. Each node *k* in the tree is associated with two sets: S_k, and I_k . S_k contains candidate sequence-extended sequences of node *k* and I_k contains candidate itemset-extended sequences of node *k.*

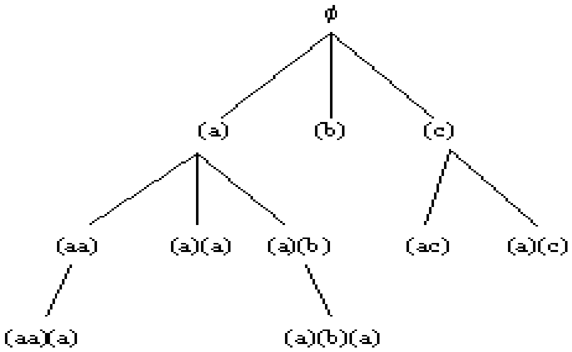


Figure 2. A sample lexicographic sequence tree.

The sample lexicographic sequence tree for the items a, b and c is shown in Fig. 2. The root of tree contains the null sequence and each lower level n has n-sequences. Each element in the tree is produced using either a sequence-extended sequence or an itemset-extended sequence. Our algorithm CSPAM traverses the lexicographic sequence tree in depth-first manner. The support of each sequence-extended child and each itemset-extended child is checked at each node. We accumulate a sequence α if its support is greater than or equal to minimum support and repeat DFS recursively on α. We do not repeat DFS on α if the support of α is less than minimum support.

C. Pruning

To improve the performance of our algorithm, we use sequence merging and sub-sequence pruning methods during the mining process to prune the search space and speed up mining.

Sequence merging: Let α be a frequent sequence. If every transaction containing sequence α also contains sequence β but not any proper superset of β, then α U β forms a frequent closed sequence and there is no need to search any sequence containing α but no β.

Sub-sequence pruning: Let α be the frequent sequence currently under consideration. If α is a proper subset of an already found frequent closed sequence β and sup(α) = sup(β), then α and all of α's descendants cannot be frequent closed sequences and thus can be pruned.

D. Candidate Generation

In this section, we first discuss sequence extension step processing and then itemset extension step processing to generate candidates using the bitmap representation.

Assume we have bitmaps *B*(*s*) and *B*(*i*) for sequence *s* and item *i* respectively. The sequence extension step on *s* using *i* will append the itemset (i) to *s*. If the bitmap for the new sequence, *B*(s_n), has a bit with value *one* then the corresponding itemset *m* must contain *i*, and all other itemsets in s_n should appear before *m*. We first produce a bitmap from *B*(*s*) such that all bits less than or equal to *m* are set to *zero*, and all bits after *m* are set to *one*. We label this bitmap as a *transformed bitmap*. We then *AND* the transformed bitmap with the item bitmap. The resultant bitmap is exactly the bitmap for the generated sequence.

The itemset extension step on *s* using *i* will create a new sequence s_n by appending item *i* to the last itemset of *s*. If the bitmap for the new sequence, *B*(s_n), has a bit with value *one* then the corresponding transaction *m* must contain the last itemset in s_n and all other itemsets in s_n should appear in transactions before *m*. Consider the resultant bitmap *B*(s_r) produced by *ANDing B*(*s*) and *B*(*i*). If bit *m* in *B*(*s*) is *one* and bit *m* in *B*(*i*) is also *one* then the bit *m* in *B*(s_r) is set to *one*. For bit *m* of *B*(s_r) to be *one*, the transaction *j* that corresponds to bit *m* should have both the last itemset in *s* and the item *i*. All other itemsets of *s* should present in transactions before *j*. Therefore *B*(s_r) is exactly the bitmap for the generated sequence.

E. *Closure checking scheme*

We have designed an efficient closure checking scheme that uses a hash index to accelerate the closure checking. The Hash index is used to maintain the set of closed sequences mined so far in memory. We use support of a sequence as hash key.

For each new frequent sequence, we have to do two kinds of closure checking such as superset-checking and subset-checking. The superset-checking tests if this new frequent sequence is a superset of some already found closed sequence candidates with the same support. The subset-checking tests if the newly found sequence is a subset of an already found closed sequence candidate with the same support.

In case of superset-checking, we verify whether the current sequence S_c subsumes another already found closed sequence or not. If the current sequence S_c subsumes another already found closed sequence S_a then they must have the following relationships: (1) S_c and S_a have the same support and (2) S_a is a part of S_c. If S_c subsumes S_a then S_a is replaced with S_c in the hash table.

In case of subset-checking, we verify whether the current sequence S_c can be subsumed by another already found closed sequence or not. If the current sequence S_c can be subsumed by another already found closed sequence S_a then they must have the following relationships: (1) S_c and S_a have the same support and (2) S_c is a part of S_a. If S_c is subsumed by S_a then S_c is not inserted into the hash table. If S_c cannot be subsumed by any other already found closed sequence then S_c is a closed sequence and it is inserted into the hash table.

F. *Algorithm*

```
Algorithm: CSPAM
Input: A sequence database SD and minimum support min_sup.
Output: The complete set of closed sequential patterns.
1. Remove infrequent items and empty sequences in SD.
2. Scan the database and construct vertical bitmap for each item in the database.
3. Initialize the bitmaps by setting the bits corresponding to the sequences.
4. Construct lexicographic sequence tree.
5. Perform depth first search on lexicographic sequence tree.
6. Perform sequence extension step and itemset extension step at each node in the lexicographic sequence
   tree and adjust bitmaps.
7. Apply sequence merging and sub-sequence pruning methods to reduce the search space.
8. Perform closure checking using hash index to eliminate nonclosed sequential patterns.
```

Figure 3. CSPAM Algorithm.

IV. PERFORMANCE EVALUATION

In our experiments we used the MSNBC dataset. It is a click stream data obtained from the UCI repository. The original dataset contains 989,818 sequences. Here the shortest sequences have been removed to keep only 31,790 sequences. The number of distinct items in this dataset is 17. The average number of itemsets per sequence is 13.33. The average number of distinct item per sequence is 5.33. The characteristics of the dataset are given in Table 2.

TABLE II. CHARACTERISTICS OF THE DATASET

S. No.	Characteristic	Value
1	No of sequences	31790
2	No of distinct items	17
3	Average no of itemsets per sequence	13.33

The experiments are conducted on a 2GHz Intel Core2 Duo processor with 1GB main memory running Windows XP. The algorithm is implemented in Java and it is executed using different support values on MSNBC dataset to find out closed sequential patterns. The Fig. 4 shows the performance comparison between FMCSP and CSPAM algorithm. Our proposed algorithm CSPAM runs faster than FMCSP.

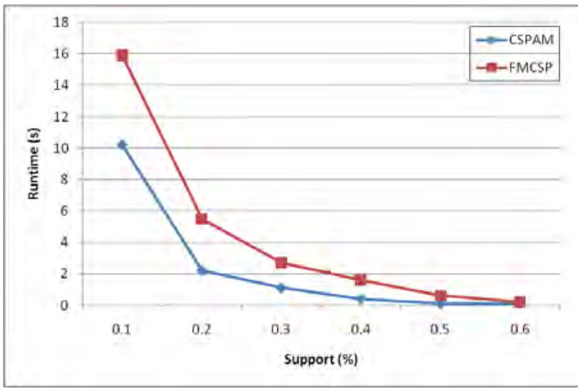


Figure 4. Performance Comparison.

V. CONCLUSION

In this paper, we propose an efficient algorithm CSPAM for mining closed sequential patterns in large data sets by integrating the best qualities of an efficient sequential pattern mining algorithm SPAM and an efficient closed itemset mining algorithm closet+. The closed sequential pattern mining has the same expressive power of sequential pattern mining and also produces more compact result set. Our algorithm CSPAM outperforms FMCSP by an order of magnitude. Other interesting research problems that can be pursued include parallel mining of closed sequential patterns and mining of structured patterns.

REFERENCES

[1] R. Agrawal and R. Srikant, “Mining sequential patterns,” Proc. Int’l Conf. Data Engineering (ICDE ’95), pp. 3-14, Mar. 1995.

[2] R. Srikant and R. Agrawal, “Mining sequential patterns: Generalizations and performance improvements,” Proc. Int’l Conf. Extending Database Technology (EDBT ’96), pp. 3-17, Mar. 1996.

[3] J. Han, J. Pei, and Y. Yin, “Mining frequent patterns without candidate generation,” Proc. ACM-SIGMOD Int’l Conf. Management of Data (SIGMOD ’00), pp. 1-12, May 2000.

[4] J. Han, J. Pei, B. Mortazavi-Asl, Q. Chen, U. Dayal, and M.C. Hsu, “FreeSpan: Frequent pattern-projected sequential pattern mining,” Proc. ACM SIGKDD Int’l Conf. Knowledge Discovery and Data Mining (SIGKDD ’00), pp. 355-359, Aug. 2000.

[5] M. Zaki, “SPADE: An efficient algorithm for mining frequent sequences,” Machine Learning, vol. 42, pp. 31-60, 2001.

[6] J. Ayres, J. Gehrke, T. Yiu, and J. Flannick, “Sequential pattern mining using a bitmap representation,” Proc. ACM SIGKDD Int’l Conf. Knowledge Discovery and Data Mining (SIGKDD ’02), pp. 429-435, July 2002.

[7] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M. Hsu, “PrefixSpan : Mining sequential patterns efficiently by prefix-projected pattern growth,” Proc. Int’l Conf. Data Engineering (ICDE ’01), pp. 215-224, Apr. 2001.

[8] X. Yan, J. Han, and R. Afshar, “CloSpan: Mining closed sequential patterns in large databases,” Proc. SIAM Int’l Conf. Data Mining (SDM ’03), pp. 166-177, May 2003.

[9] P. Tzvetkov, X. Yan, and J. Han, “TSP: Mining top-k closed sequential patterns,” Proc. IEEE Int’l Conf. Data Mining (ICDM ’03), pp. 347-354, Dec. 2003.

[10] S. Cong, J. Han, and D.A. Padua, “Parallel mining of closed sequential patterns,” Proc. ACM SIGKDD Int’l Conf. Knowledge Discovery and Data Mining (SIGKDD ’05), pp. 562-567, Aug. 2005.

[11] J. Wang, J. Han, and Chun Li, “Frequent closed sequence mining without candidate maintenance,” IEEE Trans. Knowledge and Data Eng., vol. 19, no. 8, pp. 1042-1056, Aug. 2007.

[12] J. Pei, J. Han, and R. Mao, “CLOSET: An efficient algorithm for mining frequent closed itemsets,” Proc. ACM SIGMOD Workshop Research Issues in Data Mining and Knowledge Discovery (DMKD ’00), pp. 21-30, May 2000.

[13] M. Zaki and C. Hsiao, “CHARM: An efficient algorithm for closed itemset mining,” Proc. SIAM Int’l Conf. Data Mining (SDM ’02), pp. 457-473, Apr. 2002.

[14] J. Han, J. Wang, Y. Lu, and P. Tzvetkov, “Mining top-k frequent closed patterns without minimum support,” Proc. IEEE Int’l Conf. Data Mining (ICDM ’02), pp. 211-218, Dec. 2002.

[15] J. Wang, J. Han, and J. Pei, “CLOSET+: Searching for the best strategies for mining frequent closed itemsets,” Proc. ACM SIGKDD Int’l Conf. Knowledge Discovery and Data Mining (SIGKDD ’03), pp. 236-245, Aug. 2003.

[16] Nancy P. Lin, Wei-Hua Hao, Hung-Jen Chen, Hao-En Chueh and Chung-I Chang, “Fast mining of closed sequential patterns,” WSEAS Transactions on Computers, vol. 7, no. 3, Mar. 2008.