Efficient Approach for Multi Dimensional Closed Sequence Mining

Hamsaa Sayeekrishnan
Zoho Corporation
Chennai, India
hamsaa.sk@zohocorp.com

Jaisuriya
Zoho Corporation
Chennai, India
jaisurya.sg@zohocorp.com

Egappan
Zoho Corporation
Chennai, India
egappan.re@zohocorp.com

Deepan
Zoho Corporation
Chennai, India
deepan.mn@zohocorp.com

Prathap Manohar Joshi
Zoho Corporation
Chennai, India
prathap.joshi@zohocorp.com

Ramki
Zoho Corporation
Chennai, India
ramki.r@zohocorp.com

***Abstract*—Sequence mining is the discovery of useful ordered patterns from the data. A good example of sequence mining would be analysis of customer purchases in an e-commerce site. Each sequence is usually associated with a set of features or dimensions. As an example, in an e-commerce site , each purchase is associated with a customer, place, timestamp etc.While Sequence mining can give us frequently bought products, Multi dimensional sequence mining can give us a more useful insights on which customer, at what place and at what time purchased a particular sequence of products.**

Previous works on multi dimensional sequence mining is based on extracting the full set of sequences along with multi dimensional patterns. Extracting the full set of sequence is a disadvantage, since mining of long sequences will further generate huge number of sub-sequences which is costly in terms of time and space. In this paper, we propose an unique and scalable algorithm for mining sequence patterns that are not full set of sequences, but closed, along with their multi dimensional information. Our approach integrates a closed sequence mining algorithm and a multi dimensional pattern mining algorithm - CM-ClaSP and Bottom Up Computation. We have mentioned many ways to combine the sequence mining with multi dimensional analysis methods and discuss a novel and scalable approach for the same. Furthermore, we also performed extensive experiments to understand the scalability and performance of the algorithm, which show that the algorithm is scalable, fast and efficient.

***Keywords*—Closed sequence mining, multi dimensional pattern mining**

I. Introduction

Sequential pattern mining discovers frequent sub-sequences from database containing sequences. It has a wide variety of applications including market basket analysis (analysis of customer purchase sequences while shopping), web usage mining (analysis of user action on websites to get browsing patterns and to improve user experience), healthcare data analysis (analysis of patient medical records to understand sequence of medical events), Genomic Data Analysis (identifying patterns in DNA or protein sequences to understand genetic relationship) and many more.

But the patterns found in sequential pattern mining lack specificity and focus, as they are generic in nature. For example: if sequence mined in market basket analysis is bread and milk, this maybe more suitable for people beyond 40 and bread and cheese maybe more suitable for younger generation. Hence, if sequence pattern can be linked with multi-dimensional features, the resulting sequences can be very useful [1].

A recent study by Pinto et al, explores multi-dimensional sequential pattern mining by integrating sequence mining algorithm (Prefix-Span) and multi-dimensional pattern mining (Iceberg-cube computation: bottom up computation-BUC) [1]. Algorithms like Prefix-Span show good performance only on short sequences and high minimum support threshold. Hence for long sequences and low minimum support the prefix-span algorithm is not efficient. (*Minimum support is a threshold to decide if a particular pattern is useful or not.*) Further, prefix-span does not perform well on dense datasets [2]. This problem can be solved using closed sequences.(*A sequence is closed if there is no super-sequence with the same support in the database*). Gomariz et al, proposed an efficient algorithm for mining frequent closed sequences called Closed sequential patterns algorithm (CM-ClaSP) , which outperforms algorithms like prefix-span and SPADE.

In this paper we propose a novel efficient algorithm for multi-dimensional closed sequence pattern mining. We combine CM-ClaSP and BUC (iceberg-cube computation) - two efficient algorithms on sequence mining and multi-dimensional analysis. Furthermore, we are proposing a novel optimized approach to integrate the closed sequence mining algorithm and by multi dimensional pattern mining , which is memory optimized and fast.

The remaining paper is organized as follows. Section 2 introduces all the basic concepts of closed sequential pattern mining and notations used in this paper. In section 3 we present related works regarding closed sequence mining and discuss the algorithm we use. In section 4 we present related works regarding multi-dimensional analysis and discuss the algorithm we use. In section 5, we discuss three ways of combining the above two algorithms and arrive at an optimized solution. In

section 6, we present the performance study and lastly discuss the conclusions.

II. Problem Setting

*1) **Item and Itemsets**:* Let $I = \{i_1, i_2, ..., i_n\}$ represent a set of items. An itemset X is defined as a subset of items, denoted as $X \subseteq I$. A sequence is an ordered list of itemsets, represented as $(s_1, s_2, ..., s_l)$, where s_j is an itemset, i.e., $s_j \subseteq I$ for $1 \leq j \leq l$. Each s_j is also referred to as an element of the sequence, denoted as $(x_1, x_2, ..., x_m)$, where x_k is an item, i.e., $x_k \in I$ for $1 \leq k \leq m$. For simplicity, brackets are omitted when an element contains only one item, so x is used instead of (x). An item can appear at most once in an element, but it can occur multiple times in different elements of the sequence.

*2) **Support of a Sequence**:* The support of a sequence a in a sequence database S is the count of tuples in the database containing a, denoted as $support(a) = |\{(sid, s) \mid (sid, s) \in S \wedge a \subseteq s\}|$. This can also be expressed as $support(a)$ when the context makes the sequence database clear.

*3) **Minimum Support**:* Given a positive integer $min\_support$ as the support threshold, a sequence a is considered a sequential pattern in the sequence database S if it is contained in at least $min\_support$ tuples, i.e., $support(a) \geq min\_support$. A sequential pattern with length l is referred to as an l-pattern.

*4) **Frequent Item**:* An item of an itemset is frequent if the support of the item is greater than the minimum support value.

*5) **Closed and Maximal Sequences**:* If there are no frequent itemset Y such that $X \subset Y$, the itemset is said to be maximal. From an maximal itemset , we can recover all the other frequent subsets by enumerating over the itemset. Another representation of frequent itemset is closed. If there are no frequent subset Y such that $X \subset Y$ and $support(X) = support(Y)$

*6) **Sequence Database**:* A sequence database S is a set of tuples (sid, s), where sid is an identifier for the sequence and s is a sequence. If a tuple (sid, s) contains a sequence a, it means a is a subsequence of s, denoted as $a \subseteq s$.

*7) **Multi Dimensional Sequence Database**:* A multi-dimensional sequence database follows the schema $(RID, A1, ..., Am, S)$, where RID is a primary key, A1, ..., Am are dimensions, and S belongs to the domain of sequences.

III. Closed Sequence pattern Mining

Agarwal et al proposed the first algorithm in mining frequent sequences which was apriori based *(any super pattern of an infrequent pattern cannot be frequent)*, where candidate sequences are generated in an iterative manner and pruned using support. The first scan finds all the frequent items , called seed set, which is used to generate potential patterns called candidate sequences. Each candidate sequence contains one more item than the seed set. The candidate sequence becomes the seed set for the next scan and the process is repeated till we find the frequent sub-sequences. Though the number of iterations has reduced when compared to brute force, the algorithm requires multiple scans to generate candidate sequences and may lead to the generation of large candidate sequences [3].

TABLE I
Sequence Data

Sequence-ID	Sequences
1	$< (ab)c(fg)g(e) >$
2	$< (ad)c(b)(abef) >$
3	$< (a)b(f)(e) >$
4	$< (b)(fg) >$

Considering the disadvantages, another algorithm was proposed by Han et al., to reduce the candidate sequence generation , called Prefix-Span algorithm . In the first scan, the algorithm finds the single item frequent sequences with pruning of those whose frequency is less than $min\_support$. For each of the items, a projected database is formed, which will consist of the post-fix sequences for the prefix (item), upon which new frequent items are identified. This process is repeated recursively with the help of DFS and new patterns are formed by concatenating the frequent items to the prefix. This algorithm performs well on large databases where the item-set is short [20].

Another algorithm to reduce the generation of candidate sequences in Apriori based algorithm, is by using a vertical database format of the original database. This algorithm is called SPADE (Sequential pattern discovery using equivalence classes). The primary stages involve calculating the frequent 1-sequences and 2-sequences, breaking down the sequences into parent equivalence classes based on prefixes, and then systematically listing all additional frequent sequences using either Breadth-First Search (BFS) or Depth-First Search (DFS) exploration within each equivalence class. The SPADE algorithm unlike previous algorithms, makes only 3 scans of database - one for frequent 1-sequences , one for frequent 2-sequences and last for generating all other frequent sequences. But since there are a lot of candidate sequences being generated , we can further optimize this algorithm [8].

All these algorithms mine frequent sequences but none of them mine closed frequent sequences. CloSpan (Closed sequential pattern mining algorithm) was proposed by Yan et al. This algorithm mines closed frequent sequence , along with pruning methods such as backward sub-pattern and backward super-pattern are used , where patterns will be absorbed or merged based on whether they are closed or not. CloSpan also generates candidate sequences in form of prefix tree. CloSpan performance is claimed to be better than prefixSpan algorithm as it mines only closed frequent sequences [5].

A new data structure , called co-occurence map was defined by Fournier-Viger et al, to store co-occurence information. This will help in pruning of candidate sequences in all the above algorithm. $cmap_i$ stores every item that succeeds each item by i-extension atleast $min\_support$ times. $cmap_s$ stores every item that succeeds each item by e-extension atleast

TABLE II
$cmap_i$ TABLE

item	is succeeded by (i-extention)
a	$\{b\}$
b	$None$
c	$None$
e	$None$
f	$\{g\}$
g	$None$

TABLE III
$cmap_s$ TABLE

item	is succeeded by (s-extention)
a	$\{b, c, e, f\}$
b	$\{e, f, g\}$
c	$\{e, f\}$
e	$None$
f	$\{e, g\}$
g	$None$

$min\_support$ times. This can be used in pruning of the candidate sequence generated based on $min\_support$. This technique can be incorporated in algorithms such as prefix-span and SPADE. Using cmap, CM-CloSpan, CM-SPADE and CM-prefix span algorithms were introduced , where cmap was used for pruning [10].

Gomariz et al , introduced an algorithm called ClaSP (Closed sequential pattern algorithm) which mines closed frequent patterns using vertical database format. In the first scan, the algorithm finds frequent-1 sequences. Using Depth First Search Pruning method , the algorithm recursively iterates over the tree and generates frequent closed candidates. Finally, once the candidates are generated , the algorithm removes non-closed candidates and returns frequent closed sequences. The pruning is the algorithm is based on CloSpan - backward sub pattern checking and backward super pattern checking. If the pattern is a sub-pattern of the previous pattern found , then that particular branch will not be searched. If the pattern is a super-pattern of a pattern, previously found, then the branches of that pattern are merged with the current one. The main difference of ClaSP with SPADE is that in SPADE we do not check whether a sub-tree can be skipped and we do not remove the non-closed sequences. The performance of this algorithm when compared with SPADE and CloSpan, ClaSP outperforms CloSpan and SPADE. An upgrade of ClaSP is CM-ClaSP where we use cmap for pruning, similar to CM-Spade. Performance study was conducted on these algorithms on different datasets - Leviathan , Sign , Snake, FIFA , BMS and Kosarak10k. It was found that CM-ClaSP outperformed all the other algorithms in closed sequence mining in Leviathan , Snake, BMS and Kosarak10k [2].

Based on the performance and our desired output to mine closed sequences, we chose the CM-ClaSP algorithm to mine frequent patterns. This will be the first step in our multi dimensional sequential pattern mining algorithm. The next section explains the multi dimensional pattern mining.

IV. Multi-Dimensional Pattern Mining

In relational database , a data cube can be used to store data of a particular interest. Only those cells in a data-cube that meet an aggregate condition is called an iceberg cube. Based on the aggregate condition, some cubes are stored and some are not. With the help of iceberg cube, only those values which support in decision making queries can be stored. Iceberg cube computation mainly consists of two methods - top-down approach and bottom up approach. Findlater et al, proposed three iceberg cube algorithms - Bottom up computation (BUC), top down computation (TDC) and top down computation with pruning (TDCP) [21].

Bottom up computation first counts the frequency of first attribute and partitions the database for those values which are frequent, only after sorting them. The algorithm proceeds to the next attribute, among the frequent tuples , prunes based on frequency and iteratively goes to all the attributes and returns the tuple of attributes that are frequent. This similar process is followed starting with second attribute , third attribute and so on.

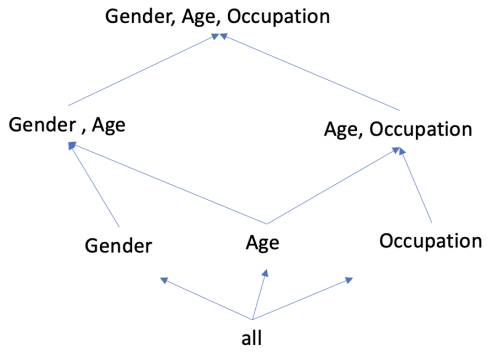

Fig. 1. Processing tree for BUC algorithm.

Figure 1 displays the BUC algorithm's processing tree for a database with three dimension: A, B and C. After analysing dimension A, the algorithm divides the database into groups based on how frequently dimension A is used. Each group is then sorted by the next dimension, B. BUC counts the combinations of dimension A and B inside each partition. Once more, the database is partitioned based on frequent combinations of AB and sorted by dimension C after the frequent combinations are identified. The method moves on to the next node and traverses back down the tree if no frequently occurring combinations are detected.

On the contrast the top down computation , a lattice with all possible combination is formed as shown in figure 2. The algorithm traverses from top of the lattice and based on the frequency it creates nodes in the processing tree for TDC algorithm. The advantages of BUC algorithm over TDC algorithm is that BUC algorithm is more efficient in pruning , whereas in TDC , first a lattice is created upon which pruning is done. Also BUC is more memory efficient when compared to TDC. Based on the above research, we considered

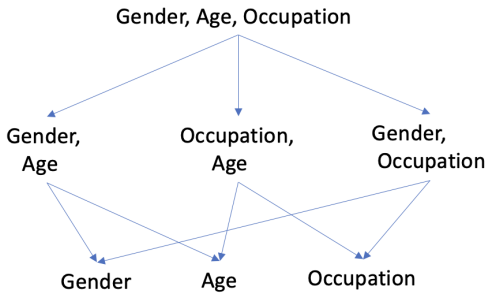

Fig. 2. Lattice for TDC algorithm.

BUC algorithm for multi dimensional pattern mining. In the next section we discuss our algorithm integration for closed sequence mining to multi dimensional pattern mining.

V. Combining sequence mining and multi dimensional pattern mining

Let us use an example to understand the working flow of the algorithm. Lets consider a database containing details of customer (gender , age, etc) who purchased products(category, name, dimensions, weight, price, etc) and their purchase details(purchase quantity, timestamp, etc). For each customer we can extract a sequence of their product purchases. On this sequence we will apply CM-ClaSP - closed sequence mining algorithm. We will get a frequent closed sequences as an output. In order to extract the features of the customers who bought it and features of the product to extract the dimensions, we propose three approaches , after this step:

- Our first approach is a brute force method to find set of customers that bought the products present in the closed sequence mined by our CM-ClaSP algorithm using database query and similarly extract the features of the customers using query. Finally, we will have a set of unique features/dimensions for that particular product sequence. The disadvantage of this method is for each pattern mined , the database has to be scanned, more the patterns mined , more time consuming the process is and memory inefficient.
- Another better approach would be to use the idea of projected database, similar to that of prefix-span algorithm. For each unique sequence that is mined , we form a projected database , containing postfix sequences of the prefix which is the product sequence. We then prune the items in the projected database that is not frequent. Following this process , we can get a set of features/dimensions which are unique to the particular product sequence mined. This algorithm reduces the number of data retrieval and database scans when compared to the brute force method. But the disadvantage of this method would be the generation of the projected database every time for each closed sequence, which is again memory inefficient.
- Our final approach to extract all features/dimensions, is with the help of inverted index. Inverted index is a way of storing / accessing data , on the basis of referencing. We can understand the concept of inverted index with the help of example shown in table 4. Consider a mapping from the product to the list of customers who bought it and the order of purchase of a particular product by the customer. Lets say , a frequent closed sequence mined is AB . The customers and product purchase index for A is extracted from table 4. Since the next purchase is B , for combining attributes , two conditions are checked , whether the customer who has purchased A is purchasing B and the product purchase index of B is more than that of A , implying the customer has purchased B after purchasing A. From the table 4 , we can extract the following customers who have purchased AB (order is conserved) , i.e. cus_001 and cus_003. For these customers , we extract their attributes from the original database and finally arrive at a list of attributes for each customer as shown in table 5. For each of the customer attributes for an unique closed product sequence , we apply the BUC algorithm to determine the frequent attributes set unique to a particular product sequence.

TABLE IV
Inverted Index for customer purchase

Products	Customers	Product Purchase index
A	cus_001	1,3
	cus_002	2
	cus_003	1
B	cus_001	2
	cus_003	5
C	cus_001	3,4
	cus_005	1,2
	cus_009	2
D	cus_003	1

TABLE V
Attributes of customer for unique product sequence

Product sequence	Customer Attributes
AB	Male , 50, Manager Female, 50, Homemaker
BC	Male, 50, Manager Male, 25, Developer Female , 50, Doctor

TABLE VI
Frequent product sequence

Product sequence	Frequent Customer Attributes
AB	50
BC	Male, 50

VI. Performance Study

In this section, we summarise the performance of our algorithm. For our experiments , we used the movie lens

dataset. The 5-star rating and free-text tagging activity from MovieLens, a movie recommendation service, are described in this dataset (ml-25m). Across 62423 films, it has 25000095 ratings and 1093360 tag applications. 162541 users generated these data between January 9, 1995, and November 21, 2019. The creation date of this dataset is November 21, 2019 [22].
Figure 3 represents scalability of our algorithm over minimum support. We have considered the total transaction data with an average of 30 item per sequence and 4 features. It can be seen that the model scales well , as the minimum support increases, with an average run time of 0.598 sec.
To understand the scalability of model vs number of sequence, we recorded the runtime by increasing the number of sequences for a constant minimum support of 0.2. As the dataset gets larger and larger , we can see that our model scales well with an average run time of 0.136 sec.
In Figure 5 , we have plotted the scalability of model with increase in number of features. For this experiment, we recorded the runtime with increasing number of features, for number of sequence as 500, a minimum support of 0.2 and a cardinality per feature as 4. The model has an average runtime of 0.236 sec.
Another experiment was conducted to determine the scalability vs cardinality, for number of sequences as 500 and number of features as 5 (figure 6). The runtime is almost constant with average runtime of 1.42 sec. From the above experiments, we can conclude that the model is scalable as the variability of runtime for each experiment seems to be very less. The model is fast and efficient with an average runtime of 0.59 sec.

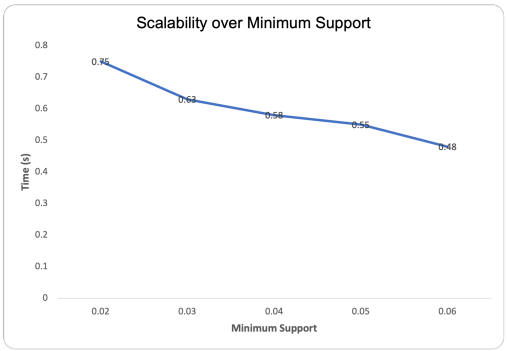

Fig. 3. Scalability over Support for total transaction with an average of 30 item per sequence and 4 features

VII. Conclusion

In this paper, we are proposing a novel, fast and scalable algorithm for multi dimensional closed sequential mining. The uniqueness of our algorithm is in mining for closed frequent sequences instead of frequent sequences and using the concept of inverted index to mine multi dimensional patterns for each of the frequent closed sequence, efficiently. The algorithm first uses CM-ClaSP to extract closed sequences This algorithm is based on using vertical database format for mining patterns and pruning the sequences not frequent using CMaps and merging patterns in tree, based on whether they are super

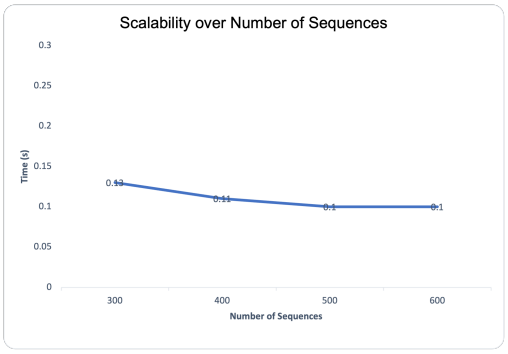

Fig. 4. Scalability over Number of Sequences for a minimum support = 0.2

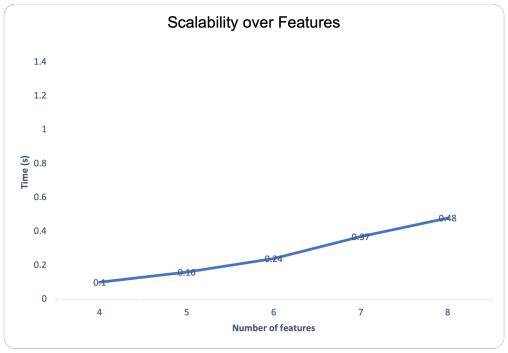

Fig. 5. Scalability over features with number of sequence = 500 , minimum support = 0.2 and cardinality = 4

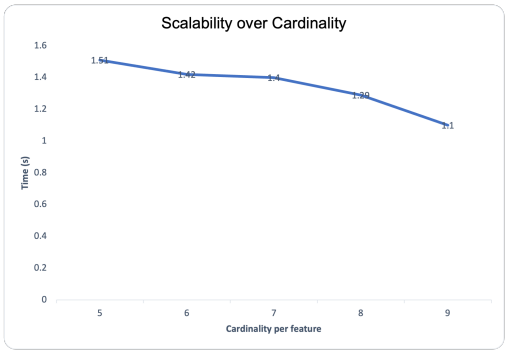

Fig. 6. Scalability over Cardinality for sequence length = 500 and number of features = 5

pattern or a sub pattern of another already mined sequence. We then propose three ways to integrate the closed sequence mining algorithm with the multi dimensional pattern mining algorithm . For multi dimensional pattern mining algorithm, we use the BUC, which is the bottom up computation - an algorithm in the iceberg cube computations. This algorithm repeatedly scans the database and partitions the database based on the frequency of the tuple values in an dimension. In order to combine the two, we propose the concept of inverted index table which will map, each unique item in the sequence to its feature and the order of occurrence of the item in the sequence. With the help of this, we can extract feature sets for each closed frequent sequence and apply the BUC algorithm on

each of the sets to mine frequence multi dimensional patterns. After performing experiments to understand the scalability, we can conclude that our model is scalable over support, cardinality per feature, number of sequences and number of features.

REFERENCES

[1] H. Pinto, J. Han, J. Pei, K. Wang, Q. Chen, and U. Dayal, "Multi-dimensional sequential pattern mining," Jan. 2001, doi: https://doi.org/10.1145/502585.502600.

[2] A. Gomariz, M. Campos, R. Luis, and B. Goethals, "ClaSP: An Efficient Algorithm for Mining Frequent Closed Sequences," *Lecture Notes in Computer Science*, pp. 50–61, Jan. 2013, doi: https://doi.org/10.1007/978-3-642-37453-1_5

[3] Agrawal, R. & Srikant, R. (1995). Mining sequential patterns. In 11th Intl. Conf. on Data Engineering

[4] M. J. Zaki, *Machine Learning*, vol. 42, no. 1/2, pp. 31–60, 2001, doi: https://doi.org/10.1023/a:1007652502315.

[5] X. Yan, J. Han, and R. Afshar, "CloSpan: Mining: Closed Sequential Patterns in Large Datasets," *Proceedings of the 2003 SIAM International Conference on Data Mining*, May 2003, doi: https://doi.org/10.1137/1.9781611972733.15.

[6] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[7] K. Beyer and R. Ramakrishnan, "Bottom-up computation of sparse and Iceberg CUBE," *ACM SIGMOD Record*, vol. 28, no. 2, pp. 359–370, Jun. 1999, doi: https://doi.org/10.1145/304181.304214.

[8] Zaki, M.J.: SPADE: An efficient algorithm for mining frequent sequences. Machine Learning 42(1), 31–60 (2001)

[9] Wang, J., Han, J., Li, C.: Frequent closed sequence mining without candidate main- tenance. IEEE Transactions on Knowledge and Data Engineering 19(8), 1042–1056 (2007)

[10] Fournier Viger, Philippe. (2014). Fast Vertical Sequential Pattern Mining Using Co-occurrence Information.

[11] Ayres, Jay & Flannick, Jason & Gehrke, Johannes & Yiu, Tomi. (2002). Sequential PAttern mining using a bitmap representation. 429-435. 10.1145/775107.775109.

[12] Fumarola, Fabio, et al. "CloFAST: Closed Sequential Pattern Mining Using Sparse and Vertical Id-Lists." Knowledge and Information Systems, vol. 48, no. 2, 20 Oct. 2015, pp. 429–463, https://doi.org/10.1007/s10115-015-0884-x. Accessed 8 Mar. 2022.

[13] Gueniche, T., Fournier-Viger, P., Raman, R., Tseng, V.S. (2015). CPT+: Decreasing the Time/Space Complexity of the Compact Prediction Tree. In: Cao, T., Lim, EP., Zhou, ZH., Ho, TB., Cheung, D., Motoda, H. (eds) Advances in Knowledge Discovery and Data Mining. PAKDD 2015. Lecture Notes in Computer Science(), vol 9078. Springer, Cham. https://doi.org/10.1007/978-3-319-18032-8_49

[14] Gueniche, T., Fournier-Viger, P., Tseng, V.S. (2013). Compact Prediction Tree: A Lossless Model for Accurate Sequence Prediction. In: Motoda, H., Wu, Z., Cao, L., Zaiane, O., Yao, M., Wang, W. (eds) Advanced Data Mining and Applications. ADMA 2013. Lecture Notes in Computer Science(), vol 8347. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-53917-6_16

[15] V. Pudi and R. Haritsa, "Generalized Closed Itemsets for Association Rule Mining," In Proc. of the 19th Int. Conf. on Data Engineering (ICDE'03), pp. 714-716, Bangalore, India, March 2003.

[16] Srikant, R., Agrawal, R. (1996). Mining sequential patterns: Generalizations and performance improvements. In: Apers, P., Bouzeghoub, M., Gardarin, G. (eds) Advances in Database Technology — EDBT '96. EDBT 1996. Lecture Notes in Computer Science, vol 1057. Springer, Berlin, Heidelberg. https://doi.org/10.1007/BFb0014140

[17] P. Songram, V. Boonjing and S. Intakosum, "Closed Multidimensional Sequential Pattern Mining," Third International Conference on Information Technology: New Generations (ITNG'06), Las Vegas, NV, USA, 2006, pp. 512-517, doi: 10.1109/ITNG.2006.41.

[18] Pitkow, J., Pirolli, P.: Mining longest repeating subsequence to predict world wide web surfing. In: Proc. 2nd USENIX Symposium on Internet Technologies and Systems, Boulder, CO, pp. 13–25 (1999)

[19] Prachi Batwara, Dr. B. K. Verma, 2014, A Time and Space Efficient Algorithm for Mining Sequential Pattern, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 03, Issue 09 (September 2014)

[20] Jian Pei et al., "Mining sequential patterns by pattern-growth: the PrefixSpan approach," in IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 11, pp. 1424-1440, Nov. 2004, doi: 10.1109/TKDE.2004.77.

[21] Findlater, Leah & Hamilton, Howard. (2003). Iceberg-cube algorithms: An empirical evaluation on synthetic and real data. Intell. Data Anal.. 7. 77-97. 10.3233/IDA-2003-7202.

[22] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4: 19:1–19:19. https://doi.org/10.1145/2827872