Research Article

International Journal of Advanced Computer Research, Vol 7(32)
ISSN (Print): 2249-7277 ISSN (Online): 2277-7970
http://dx.doi.org/10.19101/IJACR.2017.732004

Performance evaluation of top-k sequential mining methods on synthetic and real datasets

Asima Jamil[1*], Abdus Salam[2] and Farhat Amin[3]
Department of Computer Science, Shaheed Benazir Bhutto Women University, Peshawar, Pakistan[1]
Department of Computing, Abasyn University, Peshawar, Pakistan[2]
Department of Bioinformatics, Shaheed Benazir Bhutto Women University, Peshawar, Pakistan[3]

Received: 02-May-2017; Revised: 11-August-2017; Accepted: 24-August-2017

Abstract
Discovering sequential pattern from a large sequence database is an important problem in the field of sequential pattern mining, which is the well-known data mining technique. Several articles have surveyed the field of sequential pattern mining over the past few years. In those papers major focus was on improving the efficiency of algorithms by employing different techniques. However, the researchers paid less attention to consider the characteristics of the underlying data that the algorithm uses. It is very less investigated. The properties of data incredibly affect the execution of data mining algorithms. This study complemented the top-k sequential pattern mining field by providing further in depth analysis with respect to data properties and characteristics. The performance of top-k sequential pattern mining (TKS) with top-k closed sequential pattern mining (TSP), the state-of-the-art algorithm for top-k sequential pattern mining were evaluated both on synthetic and real databases. Experiments were carried out on real and synthetic datasets having varied characteristics. The impact of different parameters was investigated against the running time and memory usage analysis of each algorithm. Extensive experiments show that TKS and TSP have certain advantages and disadvantages of different types of data. Furthermore, due to the continuous addition of large amounts of data in the databases, the idea of sequential pattern mining (SPAM) is becoming popular. Various algorithms have been developed that are used for mining the sequential patterns in the data. These algorithms have proved to be more effective for smaller databases, but when the size of the database increased, their performance may decline. Hence these methods have to be amended in order to perform the mining processes in a more efficient way.

Keywords
Pattern discovery, Top-k, Data mining, Sequential pattern mining, Association rule mining.

1.Introduction
Data mining techniques are used for discovering unknown hidden patterns and to predict useful information to increase the profit of various organizations. Data mining is the extraction of implicit, previously unknown, and potentially useful information from data [1]. It involves methods at the intersection of artificial intelligence, machine learning, statistics, and database systems [2]. The methods include classification, clustering, prediction, association rule mining, sequential pattern mining. Association rule mining discovers relationships between items in a dataset irrespective of time, whereas sequential pattern mining (SPAM) considers time or order of transactions. SPM is used in various fields such as biological sequence analysis, web log click streams, medical treatment (e.g., symptoms and diseases) [3-7].

Natural disasters (e.g., earthquakes), science and engineering process, serial crime solving, telephone calling patterns, and customer purchase behaviour analysis [4].

The basic idea of SPM was first introduced by [5] for the problem of customer purchase sequence, as follows: "*Given a set consist of a number of sequences, where each sequence consists of a list of events or transactions and each event consists of a set of items, with a given a minimum support threshold, SPM is required to find all Sequential patterns (SPs), that is, the subsequence's whose occurrence frequency in the set of sequences is greater than minimum support threshold*." It is computationally complex and challenging because such mining may create large number of candidate sequences or in other words intermediate subsequence's. Since the amount of the processed data in mining SP (sequential pattern) tends to be

*Author for correspondence

huge, it is important to design methods to mine data efficiently with minimum computational cost. The main techniques used for frequent itemset mining are similar to those introduced for SPAM.

SPAM can be explained by considering the following market basket analysis example. Market basket analysis includes information that can be used to construct the purchasing history for each customer to form sequences with time stamp in ascending order. The main purpose of analysing historical shopping history data is to send promotion and advertisements to only selected valuable customers to increase the organization's profitability. The purchase history of each can be generalized as shown in the *Table 1* below, where C_1, C2, C3, C4 are customer IDs and apple, banana, carrot, diaper, egg, frozen chickens and grapes, are the various item purchases by them. They are denoted by starting letters. The items purchased together in a single transaction are enclosed in parenthesis. Each sequence includes purchase transactions, of each customer, ordered by sequence, placing transaction in the first place and last in last place with the notion of time.

Table 1 An example of customer purchase history

Customer-Id	Transaction Sequence
C_1	b(bc)(bec)c(df)e
C_2	(abd)e(b)(ab)
C_3	(fg)(ba)(de)(bc)
C_4	ge(fa)abc

As shown in the above table, each sequence consists of a number of transactions. The transaction is also called event or itemset occur at certain period of time, so the number of transactions in a sequence is ordered according with respect to time. For instance, the first sequence C_1 includes six transactions or itemsets. The first transaction in the sequence C_1 include product b then, second transaction includes two items b and c purchased at the same time, third transaction includes three items b, e and c. In each transaction order does not matter, e.g. in the second transaction bc and cb are same. Furthermore an item may appear more than once in a sequence. For instance, an item b appears more than once in the first three transactions.

2.Problem statement

This work investigated the mining top-k closed SP (TSP) [5] and efficient mining of Top-k SPAM algorithm (TKS) [6] on different test data. The data sets were divided into two categories. The first consists of synthetic data sets and real life data sets come in the second category. The performance of the two algorithms was evaluated on the bases of various key features and characteristics of the underlying data sets.

3.Literature review

The SPAM techniques are broadly categorized into three types: apriori based, pattern growth based, and top-k based. The earliest approach is based on apriori principle, called generalized SP (GSP) developed by Agrawal and Shrikant [5]. It implements a candidate generate-and-test approach using horizontal data layout. The three key features of apriori based algorithm are (*i*) candidate generate and test, (*ii*) breath first search (level wise search) and (*iii*) multiple scans of the database. Since the length of each candidate sequence grows by one at each database scan, the apriori-based method must scan the database at least k times. Because of multiple database scans, huge sets of candidate sequences are generated. These intermediate subsequence's grow exponentially to discover the required SPs. To decrease the number of sequential patterns found and find more valuable patterns, researchers have also suggested integrating constraints in sequential pattern mining [8]. Besides horizontal data formatting method used by GSP, the vertical data format can also be used. In vertical data format identifiers (id) or bitmap is used to represent transactions of items. In this format, the sequence database is converted into a vertical id list. The id list is a list of (sequence-id, timestamp) pairs indicating the occurrence time of the item in that sequence. SP discovery using equivalence classes (SPADE) technique completes database scanning in three phases using the searching approach of id-list intersections adopt a candidate generate-and-test approach using vertical data format (where the data are represented as <itemset: (sequence_ID, event_ID) >). The vertical data format can be obtained by transforming from a horizontally formatted sequence database in just one scan; however the basic methodology is breadth-first search and Apriori pruning [9]. Despite the pruning SPADE have to generate large sets of candidates in breadth-first manner in order to grow longer sequences [10-12]. However, to convert horizontal format to vertical format, requires high computational cost, which includes additional processing time and additional storage space, multiple times larger than the original sequence database [13-15].

The prefix-projected SPAM (PrefixSpan) is another algorithm based on horizontal layout, and used pattern-growth approach for SPAM [10]. In this

methodology first all frequent items are generated after scanning original database only once. After that huge set of projected databases are generated according to the number of frequent items. The projected databases are smaller databases depends on the number of frequent items generated in the first scan of the database. Each projected database is then called recursively into growing subsequence's of fragments to find the complete set of SPs. Using the divide and conquer strategy, PrefixSpan algorithm successfully generates SP. However, due to the generation of large numbers of projected databases the storage cost is high [16]. SPAM combines best features of SPADE, GSP and FreeSpan techniques [11]. It is based on bitmap vertical format representation, which is similar to the id-list approach to SPADE technique. Both SPADE and SPAM uses candidate generate and test approach using vertical format. Vertical format, the data is represented as sequence-id and transaction-id. SPAM uses depth first search strategy, completely fit into main memory. As compared to SPADE, SPAM is more efficient in terms of execution time. SPADE is more memory efficient than SPAM [15]. SPAM performs better than PrefixSpan on particular datasets having sparse and dense characteristics; SPAM is much slower than PrefixSpan with pseudo projection technique [14].

SPAM algorithms based on bitmap representation are very efficient for support counting using a vertical format by avoiding multiple scan of the database. However, the main performance bottleneck of vertical mining algorithms is that it takes lots of time in evaluating candidates that are infrequent or they are not relevant patterns [17]. To tackle this problem, co-occurrence map (CMAP) is developed. CMAPs are integrated in two state of the art vertical mining SPAM algorithms SPADE and SPAM with CM-SPADE [18]. Several efficient algorithms have been proposed for sequential data mining and one of them is a CM-SPAM algorithm [19]. All these algorithms are either directly or indirectly based on the apriori principle. Apriori property is used to trim the search space by throwing out the irrelevant patterns. For a pattern to be called a SP, it must satisfy the minimum support criterion. It is called downward closed or anti-monotonic. It is not further used in candidate generation, if the pattern support is less than minimum support threshold. In this way, large search space is pruned to make the search space more compact in order to discover SPs more efficiently [2]. In Top-k sequential pattern mining, only two techniques are available, i.e. TSP and TKS. TSP method is the first technique used for discovering the top-k closed SPs.

It is based on the PrefixSpan algorithm. TSP uses a search strategy that implements multi-pass approach. Initially, it discovers top-k frequent patterns, and then the minimum support is dynamically raised to prune the huge search space. It performs efficient closed SPAM with constraints and optimization techniques.

TKS is another algorithm for discovering the top-k SPs in a sequence database, which is based on SPAM algorithm. It has a new search strategy with an efficient support counting mechanism to generate candidates [8]. Due to vertical data representation support is performed easily using bitmaps, without scanning database again and again. The major disadvantage of TKS is that it uses large memory space to store the bitmaps and the computational cost in data conversion from horizontal to vertical data layout. In vertical bitmap representation, TKS is not efficient in terms of space utilization due to the fact that when an item is not present in an event/transaction, it requires storing zero in the bitmap to represent this element [20]. TKS uses a sequence tree to store the items Lexo-graphically. All sequences in the tree can be extended by either sequence extension or as an itemset extension. In sequence-extension sequence is generated by adding a new itemset consisting of a single item to the end of its parent's sequence in the tree. An itemset-extension includes adding an item to the last itemset in the parent's sequence [13].

4.Methodology

In this paper, we used an improved version of the old TKS algorithm. The novel TKS is based on CM-SPAM. As discussed above, it is an improved version of SPAM algorithm. So we can say that, we are using an improved version of the old technique. We compared the performance of TKS with TSP, the state-of-the-art algorithm for top-k sequential pattern mining. All algorithms were implemented in Java. The source code of all algorithms and datasets can be downloaded as part of the SPMF data mining framework. [21] Experiments were conducted on five real datasets and eight synthetic datasets having varied statistics. We investigated the impact of different parameters of the data generated on the running time of each algorithm. The parameters that we varied were the number of sequences in the dataset, average numbers of item sets per sequence, average number of distinct items per sequence by varying k parameter.

4.1Experimental evaluation

In this section, we report our experimental results on the performance of the TKS and TSP by using synthetic and real life data sets. We performed multiple experiments to evaluate the functioning of the TKS algorithm. Experiments were done on a computer with a third generation Core i5 processor running Windows 8 and 4 GB of RAM.

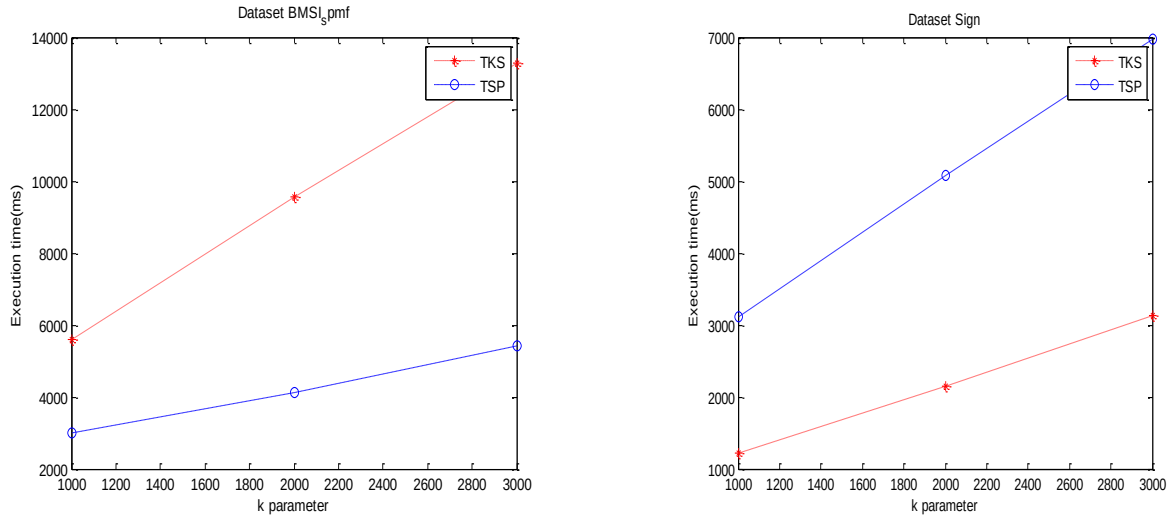
4.1.1Execution time analysis of real datasets

From this experiment, TSP performs more efficiently for BMSI_spmf and TKS perform efficiently for dataset sign. Now we discuss the statistics of real data sets. Datasets as shown in the *Table 2*.

By considering k=1000 to 3000 from BMSI_spmf. It contains 59601 number of sequences, the unique item count is 497, average number of itemsets per sequence are 2.5 and the average number of distinct items per sequence is 2.5. Similarly for sign dataset, total number of sequences is 730, number of distinct items are 267. But the average number of itemsets and distinct item per sequence is high as compared to BMSI_spmf datasets, which is 51.99. Hence the result shows that if the sequence contain a large number of itemsets and large number of distinct items, then TKS performs more efficiently as compared to TSP. But for those sequences, which contain the minimum number of itemsets and number of distinct items, TSP performs more efficiently as compared to TKS. The graphical representation for other datasets is not shown due to out of memory error for large value of k. While considering the execution time for small values of k, TKS performs more efficiently as compared to TSP as shown in *Figure 1* and *Figure 2.*

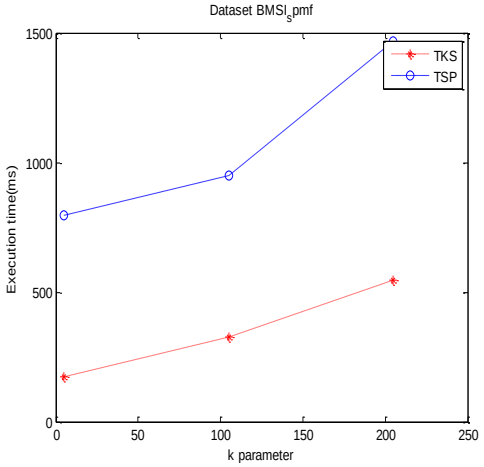
Table 2 Real life dataset statistics

Datasets	Type	Seq Count	Distinct Item	Avg Itemsets	Avg Distinct
BMSI_spmf	web click stream	59601	497	2.5	2.5
SIGN	language utterances	730	267	51.99	51.99
MSNBC	click-stream	31,790	17	13.33	5.3
LEVIATHAN	book	5834	9025	33.8	26.34
KOSARAK10K	click-stream data	10000	10094	8.1407	8.1407
SNAKE	Protein Sequences	163	20	60.61	17.84

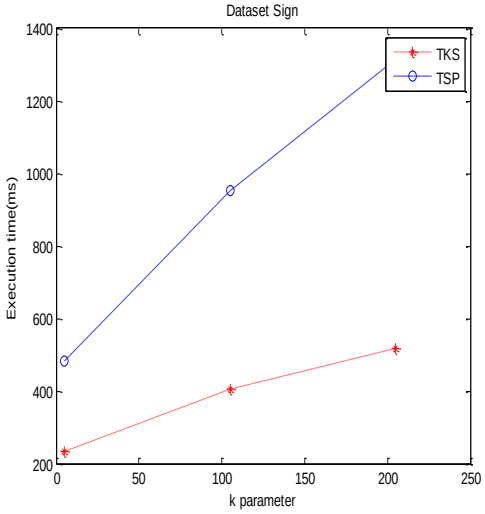


(a) (b)

Figure 1 Execution time of real datasets for large value of k



(a)



(b)

Figure 2 Execution time of real datasets for small value of k

4.1.2Memory usage analysis of real datasets

As shown in *Figure 3*, the memory results vary with changing k value for both TSP and TKS. It gives representative results for small and large value of k. For large values of k, TKS requires more space as shown in the *Figure 3* due to a large number of candidate generation. TKS is based on CM_Spam technique which consumes more memory due to a costly bitmap intersection and candidate generation. Bitmap contains ones to represent the existence of each item and zeros to represent the absence of each item.

For low value of k, TSP is more memory efficient as in case of leviathan and sign datasets. By considering the statistics of both datasets, as shown in *Table 2*. The sign datasets contain 730 numbers of sequences, the number of distinct count is 267 and the average number of distinct items per sequence is 51.99 and the average number of itemsets per sequence is also 51.99, which is high as compared to the average number of itemsets in leviathan datasets, which is 33.8 and 26.4 respectively.

Though the number of sequences in leviathan is 5834 which is high as compared to sign, which is 730.In other words, we can say TSP is more efficient in terms of memory than TKS for a minimum average number of itemsets and distinct items in each sequence.

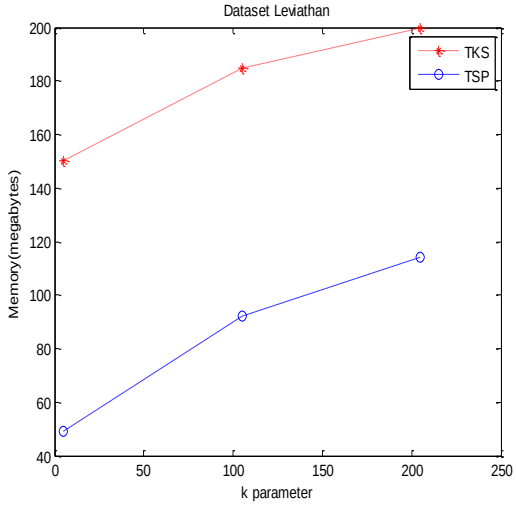


Figure 3 Results of memory usage by leviathan

Synthetic Data Generation

To test both techniques we generated numerous synthetic datasets using SPMF generate sequence database tool. There are several factors that we considered while comparing TKS against TSP. These factors are listed in *Table 3*. We also compared the performance of the algorithms by varying k value for several datasets of different sizes.

Table 3 Synthetic dataset statistics

Datasets	Sequence count	Distinct count	Avg itemsets	Avg distinct
DS100DISI232	100	232	7.4	7.4
DS50DISI644	50	644	7	20.78
DS500DISI20	500	20	7	13.65
DS600DISI30	600	30	7	15.6
DS5000DIS1000	5000	1000	7	20.8
DS5000DIS50	5000	50	7	17.5
DS5000DIS50	5000	50	100	40.90
DS5000DIS50	10000	50	7	17.5

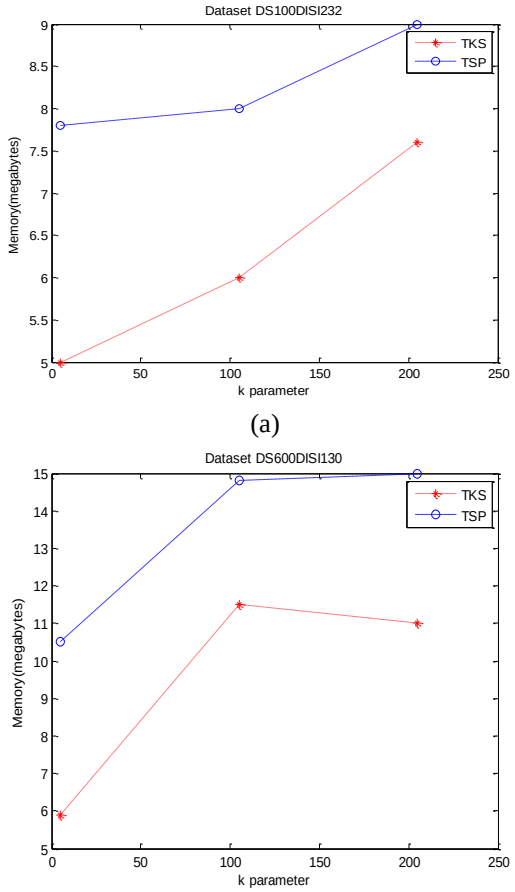


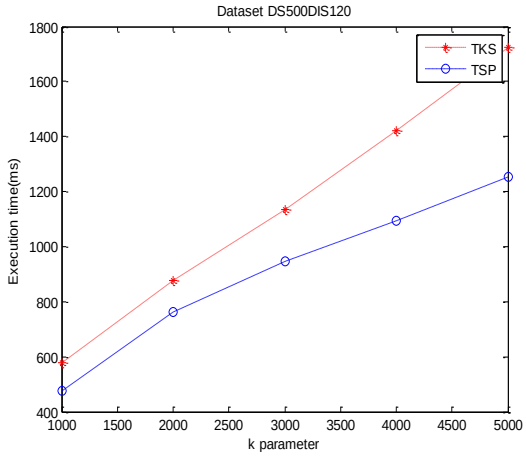
Figure 4 Memory usage of synthetic datasets

TSP outperforms TKS slightly on sequences with small length, but on long sequences TKS outperforms TSP. The results of these tests are shown in *Figure 4* and *Figure 5*. The primary reason that TKS performs so well for large datasets is due to the bitmap representation of the data for efficient counting. The counting process is critical because it is performed many times at each recursive step, and TKS handles it in an extremely efficient manner. For the short length sequence dataset, the initial overhead needed to set up and use the bitmap representation in some cases outweighs the benefits of faster counting and because of this TSP runs slightly faster for small datasets. As candidate sequences become longer, we recurs more levels down the tree and counting becomes more and more important. Overall, our runtime tests show that TKS excels at finding the frequent sequences for many different types of large dataset. Our second method of testing compared to the performance of the algorithms as several parameters in the dataset generated was varied. Experiments show that as the average number of distinct items per sequence increases, and the number of sequences decreases, the performance of TKS increases even further relative to the performance of the TSP.

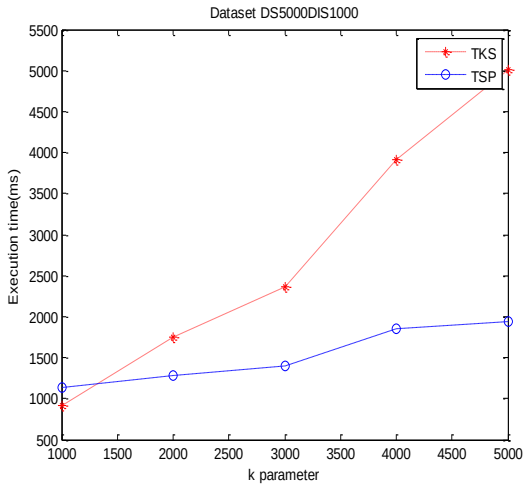
We first ran TKS and TSP on each dataset while varying k from 1000 to 5000 (typical values for a top-k pattern mining algorithm) to assess the influence of k on the execution time and the memory usage of the algorithms. Execution time results for k =1000, 2000 and 3000 are shown in *Table 4*. As it can be seen in the above table, results for the dataset, DS50DISI644 is not given, due to run out of memory during the execution of TSP algorithm. As, it is a dense dataset with long length sequence, having low sequence count and high average number of itemsets. For other datasets, both techniques give a representative result.

Table 4 Execution time of synthetic datasets for large value of k

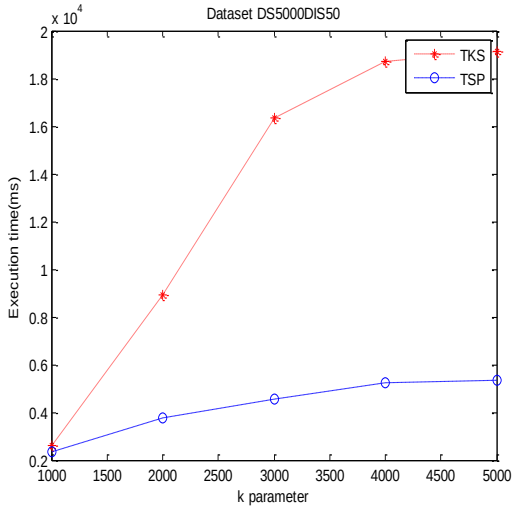
Dataset	**Algorithm**	**Execution Time(MS)**				
		k=1000	**k=2000**	**k=3000**	**k=4000**	**k=5000**
DS100DIS232	TKS	46	94	78	1140	687
	TSP	47	94	125	1984	2672
DS50DISI644	TKS	264	4000	5250	O.O.M	O.O.M
	TSP	O.O.M	O.O.M	O.O.M	O.O.M	O.O.M
DIS500DIS120	TKS	578	875	1133	1423	1723
	TSP	475	761	948	1092	1254
DIS600DIS132	TKS	922	1125	1313	1469	1578
	TSP	578	687	891	937	1000
DS5000DIS1000	TKS	923	1756	2362	3907	5007
	TSP	1141	1280	1399	1849	1935
DS5000DIS50	TKS	2593	8907	16325	18703	19110
	TSP	2360	3742	4576	5221	5350



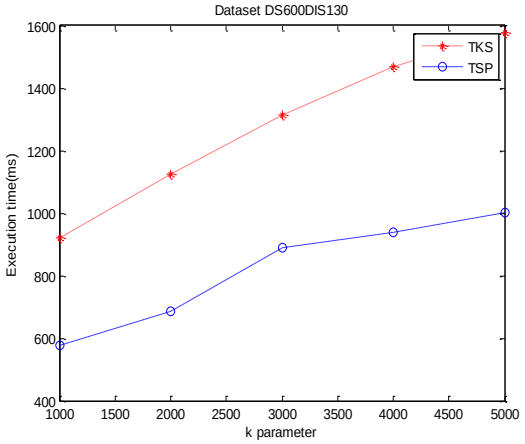
(a)



(b)

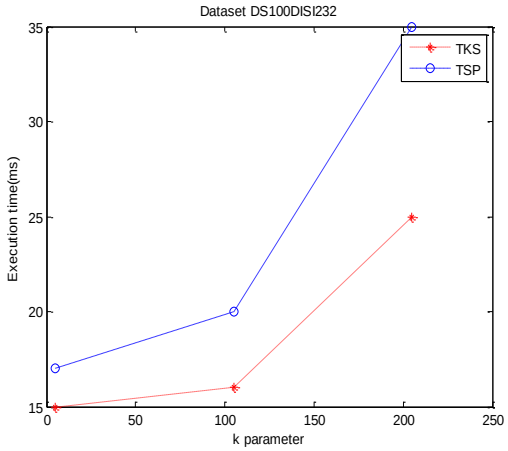


(c)

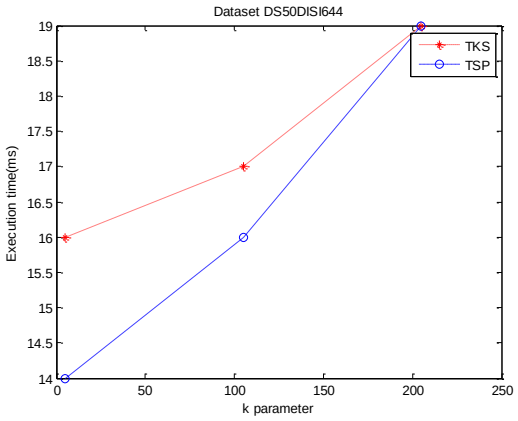


(d)

Figure 5 Execution time of Synthetic datasets for large value of k



(a)



(b)

Figure 6 Execution time of synthetic datasets for small value of k

For low value of k TKS perform more efficiently than the TSP. But in case of DS50DISI644 as shown in *Figure 6*, it performs efficiently for low values of k. It gives out of memory error for high value of k. The reason behind this is when the sequence is long, then the size of the projected database is also very large, which is costly, so TSP is not considered as an efficient technique for a dense long sequence. If the sequence is not very long, then TSP outperforms TKS, even if the database contains a number of sequences. When we talk about biological sequences, e.g.; DNA and protein sequences, then both techniques are not considered efficient. This can be proved by taking a very dense synthetic dataset, both techniques give an out of memory error due to dynamic increased with the size of the heap.

5.Conclusion and future work

The TKS techniques were compared for both small and large values of k parameter, before it was only compared for large values of k, which is not the requirement of the expected results. Secondly, it was tested on synthetic datasets for the first time, before it was only tested on real life datasets. Furthermore, the TKS is based on CM-SPAM, before it was based on SPAM technique. So the improved version of TKS technique is used in this research. In this paper, we present the performance of the TKS technique to efficiently discover Top-k sequential patterns in a given sequence database. We used an improved version of TKS based on CM-SPAM. CM-SPAM is an improved version of SPAM algorithm. Experimental results demonstrated that TKS algorithm outperforms TSP algorithm on dense datasets, but for a small length sequence, TSP outer performs than TKS even if the number of the sequence count is high. In other words, we can say that TSP outperforms TKS slightly on a short sequence datasets with maximum number of sequences, but on long sequence datasets TKS outperforms TSP by over an order of magnitude. Furthermore, experiments show that as the average number of distinct items and average number of itemsets per sequence increases, and the number of sequence decreases, the performance of TKS increases even further relative to the performance of the TSP. Due to the continuous addition of large amounts of data in the databases, the idea of sequential pattern mining is becoming popular. Several algorithms have been developed that are used for mining the sequential patterns in the data. These algorithms have proved to be more efficient for smaller databases, simply when the size of the database is increased, their performance may go down. Hence these methods have to be amended in order to do the mining processes in a safer direction.

Acknowledgment

I am thankful to Philippe Fournier-Viger for providing me the source code. I would wish to convey my earnest gratitude to Dr. Abdus Salam and Dr. Javed for the continuous support of my MS study and related research. Their guidance helped me in all the time. I am grateful to my husband for his support and encouragement. Without his support and help, it would not be possible for me.

Conflicts of interest

The authors have no conflicts of interest to declare.

References

[1] Han J, Pei J, Kamber M. Data mining: concepts and techniques. Elsevier; 2011.
[2] Witten IH, Frank E, Hall MA, Pal CJ. Data mining: practical machine learning tools and techniques. Morgan Kaufmann; 2016.
[3] Mabroukeh NR, Ezeife CI. A taxonomy of sequential pattern mining algorithms. ACM Computing Surveys (CSUR). 2010; 43(1):1-41.
[4] Dong G, Pei J. Sequence data mining. Springer Science & Business Media; 2007.
[5] Agrawal R, Srikant R. Mining sequential patterns. In proceedings of the eleventh international conference on data engineering 1995 (pp. 3-14). IEEE.
[6] Tzvetkov P, Yan X, Han J. TSP: mining top-k closed sequential patterns. Knowledge and Information Systems. 2005; 7(4):438-57.
[7] Fournier-Viger P, Gomariz A, Gueniche T, Mwamikazi E, Thomas R. TKS: efficient mining of top-k sequential patterns. In international conference on advanced data mining and applications 2013 (pp. 109-20). Springer, Berlin, Heidelberg.
[8] Mooney CH, Roddick JF. Sequential pattern mining--approaches and algorithms. ACM Computing Surveys (CSUR). 2013; 45(2):1-46.
[9] Zaki MJ. SPADE: An efficient algorithm for mining frequent sequences. Machine Learning. 2001; 42(1):31-60.
[10] Han J, Pei J, Mortazavi-Asl B, Pinto H, Chen Q, Dayal U, et al. Prefixspan: mining sequential patterns efficiently by prefix-projected pattern growth. In proceedings of the international conference on data engineering 2001 (pp. 215-24).
[11] Ayres J, Flannick J, Gehrke J, Yiu T. Sequential pattern mining using a bitmap representation. In proceedings of the eighth ACM SIGKDD international conference on knowledge discovery and data mining 2002 (pp. 429-35). ACM.
[12] Gouda K, Hassaan M. Mining sequential patterns in dense databases. International Journal of Database Management Systems (IJDMS). 2011;3(1):179-94.
[13] Salvemini E, Fumarola F, Malerba D, Han J. FAST sequence mining based on sparse id-lists. In ISMIS 2011 (pp. 316-25).

[14] Hathi KB, Varsur JA, Desai SP, Manvar SR. A performance analysis of sequential pattern mining algorithms. Journal of Emerging Technologies and Innovative Research. 2015; 2(2):397-401.
[15] Song S, Hu H, Jin S. HVSM: a new sequential pattern mining algorithm using bitmap representation. International conference on advanced data mining and applications. 2005 (pp. 455-63).
[16] Yang Z. Fast algorithms for sequential pattern mining (Doctoral dissertation).2008.
[17] Fournier-Viger P, Tseng VS. Mining top-k sequential rules. In international conference on advanced data mining and applications 2011 (pp. 180-94). Springer, Berlin, Heidelberg.
[18] Fournier-Viger P, Gomariz A, Šebek M, Hlosta M. VGEN: fast vertical mining of sequential generator patterns. In international conference on data warehousing and knowledge discovery 2014 (pp. 476-88). Springer, Cham.
[19] Chamatkar AJ, Butey PK. Comparison on different data mining algorithms. International Journal of Computer Sciences and Engineering. 2014; 2(10):54-8.
[20] Fournier-Viger P, Lin JC, Kiran RU, Koh YS, Thomas R. A survey of sequential pattern mining. Data Science and Pattern Recognition. 2017;1(1):54-77.
[21] Fournier-Viger P. SPMF : An Open-Source Data Mining Library. http://www.philippe-fournier-viger.com/spmf/. Accessed 26 March 2017.

Asima Jamil is working as a lecturer in Shaheed Benazir Bhutto Women University Peshawar. She has a teaching experience of seven years, in the field of Computer Science and Bioinformatics. She earned her M.Sc. in Computer Science from Shaheed Benazir Bhutto Women University Peshawar in 2009. She also earned her MSCS from Abasyn University, Peshawar in 2015. Her research interests include Database, Data mining and Bioinformatics.
Email: asima.jamil@yahoo.com

Abdus Salam has a distinguished career in the field of Computer Science spanning over 27 years. This includes 14 years of professional experience and 13 years of teaching experience. He earned his M.Sc in Computer Science from Quaid-e-Azam University in 1987. He also earned his Ph.D in Computer Science from International Islamic University, Islamabad, in 2011. Presently,he is working as Head of Department of Computing and Technology, Abasyn University, Peshawar, Pakistan.

Farhat Amin has a distinguished career in the field of Computational Biology/Bioinformatics spanning over 13 years of Teaching and Research experience. She earned her M.Sc in Biochemistry from University of Peshawar in 2002. She also earned her Ph.D in Biotechnology with the specialization of Biochemistry & Bioinformatics from University of Peshawar in 2011. Presently, she is working as Head of Department of Bioinformatics, Shaheed Benazir Bhutto Women University, Peshawar, Pakistan. The fields of interest are Bioinformatics, Chemistry, Biochemistry and Biotechnology.