International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868
Foundation of Computer Science FCS, New York, USA
Volume 9 – No.3, June 2015 – www.ijais.org

PrefixSpan Algorithm for Finding Sequential Pattern with Various Constraints

Pratik Saraf
M.E. Scholar
Computer Engineering Department,
Thakur College of Engineering and Technology, Mumbai

R. R Sedamkar
Professor
Computer Engineering Department,
Thakur College of Engineering and Technology, Mumbai

Sheetal Rathi
Assistant Professor
Computer Engineering Department,
Thakur College of Engineering and Technology, Mumbai

ABSTRACT
PrefixSpan (Prefix-projected Sequential pattern mining) algorithm is very well known algorithm for sequential data mining. It extracts the sequential patterns through pattern growth method. The algorithm performs very well for small datasets. As the size of datasets increases the overall time for finding the sequential patterns also get increased. The PrefixSpan algorithm is run on different datasets and results are drawn based on minimum support value. One new parameter maximum prefix length is also considered while running the algorithm. Through maximum prefix length parameter the length of prefix pattern is set which is helpful for running the algorithm on large datasets. The paper also shows the variation in time complexity and memory utilization while running the algorithm on different size of input sequential datasets.

Keywords
PrefixSpan Algorithm, Minimum support, Maximum prefix length, Time complexity, Memory utilization.

1. INTRODUCTION
The sequential pattern mining problem was first addressed by Agrawal and Srikant [1995] [1, 2]. They said that, for a given sequential database, in which each sequence consists of a list of transactions. All these transactions are ordered by transaction time and each transaction is a set of items. Sequential pattern mining is made in order to discover all sequential patterns based on user-defined minimum support. The support of a pattern is calculated through the number of data-sequences that the pattern contains.

Sequential Pattern Mining is a well known data mining technique which consists of finding sub-sequences and patterns which are appearing in a given set of sequence very often. The PrefixSpan algorithm which is proposed by Jian Pei et al. widely used to find the sequential patterns. It avoids the huge candidate sequence generation thus improvise the execution time and memory utilization.

The paper contains the results of execution of PrefixSpan algorithm on various datasets. The first section of the paper gives the brief of various sequential pattern mining algorithms. The second section deals with the objective and the scope of the result which are delivered by execution of the algorithm. The third section gives the brief about the PrefixSpan algorithm and its steps of execution. In fourth

Section, the results are drawn by executing the algorithm on different datasets.

2. RELATED WORK
Sequential pattern mining [1, 2] was first introduced by Agrawal and Srikant in 1995, and three algorithms as AprioriSome, AprioriAll and DynamicSome [1] are proposed by them. Then different parameters such as time constraints, sliding window time, and user-defined systematic, are used so as to generalise the definition of sequential pattern mining and proposed an Apriori-based, improved algorithm as GSP (Generalized Sequential Patterns). Zaki brought up SPADE [3] algorithm which was based on the equivalence of classes. It was simply the expansion of vertical data format sequential pattern mining method. Later pattern growth method come into exists. Two pattern growth algorithms were proposed by Han, which included FreeSpan [4] and PrefixSpan [5]. Compared with projected databases and subsequence connections, PrefixSpan is more efficient than FreeSpan. SPMIP (Sequential Pattern mining based on Improved PrefixSpan) algorithm [6] by LIU Pei-yu et.al and BLSPM (bi-level Sequential Pattern mining) algorithm [7] by Lian Dong and Wang hong are proposed which overcome the problem of constructing huge projected dataset in PrefixSpan algorithm.

3. OBJECTIVE OF THE WORK
The PrefixSpan algorithm is run on various datasets. The sizes of datasets are increased gradually so as to check the execution of algorithm from small datasets to large datasets. Various parameters like memory utilization, time complexity and size of projected dataset are set as benchmark for evaluating the results derived by algorithm on different datasets. This is done to get idea for improvising the existing PrefixSpan algorithm.

4. PREFIXSPAN ALGORITHM
A pattern-growth method based on projection is used in PrefixSpan algorithm [5] for mining sequential patterns. The basic idea behind this method is, rather than projecting sequence databases by evaluating the frequent occurrences of sub-sequences, the projection is made on frequent prefix. This helps to reduce the processing time which ultimately increases the algorithm efficiency.

Jian Pei et al. proposed a novel algorithm called PrefixSpan (Prefix-projected Sequential Pattern Mining) algorithm [5] which works on projection of database and sequential pattern growth. The divide and search space technique is implemented by PrefixSpan. Algorithm mines sequential patterns through following steps;

i. Find length-1 sequential patterns. The given sequence S is scanned to get item (prefix) that occurred frequently in S. For the number of time that item

occurs is equal to length-l of that item. Length-l is given by notation <pattern> : <count>.

ii. Divide search space. Based on the prefix that derived from first step, the whole sequential pattern set is partitioned in this phase.

iii. Find subsets of sequential patterns. The projected databases are constructed and sequential patterns are mined from these databases. Only local frequent sequences [8], [9] are explored in projected databases so as to expand the sequential patterns. The cost for constructing projected database is quite high. Bi-level projection and pseudo-projection methods are used to reduce this cost which ultimately increases the algorithm's efficiency.

3.1 Example

Following are some basic definitions which are necessary to understand the execution of PrefixSpan algorithm.

Sequence Dataset: It is a set of sequences where each sequence having a list of item sets.

Item set: It is an unordered set of distinct items.

Support of a sequential pattern: It is the number of sequences which are calculated by dividing the pattern occurs with the total number of sequences in the database.

Frequent sequential pattern: The sequential pattern having a support more than the *minsup* parameter which will be provided by the user.

The input of PrefixSpan is a sequence database and a user-specified threshold named minimum support (minsup).

The table shown below contains four sequences as S1, S2, S3 and S4. The first sequence, named S1, contains 7 item sets. All the items in an item set are sorted based on lexicographical order. The repetition of items in the same item set should not be occurred and that items.

Table 1. Sequence database

ID	Sequences
S1	(1), (2), (1 2) , (3), (1 3), (4 5), (6)
S2	(3 4), (3), (2 3), (1 4)
S3	(4 5), (2), (2 3 4), (3), (1)
S4	(4), (5), (1 6), (3), (2), (7), (1)

PrefixSpan discovers all frequent sequential patterns occurring in a sequence database having value greater than *minsup* value which is provided by the user.

A sequential pattern is a sequence. A sequence SA = P1, P2 ...Pm, where P1, P2... Pm is item sets is said to occur in another sequence SB = Q1, Q2 ...Qn, where Q1, Q2... Qn are item sets, if and only if there exists integers 1 <= i1 < i2... < im <= n such that P1 ⊆ Qi1, P2 ⊆ Qi2 ... Pn ⊆ Qin.

For example, if we run PrefixSpan with minsup= 0.5 and with a maximum pattern length of 50 items, 47 sequential patterns are found. The list is too long to be presented here. An example of pattern found is "(2, 3), (4)" which appears in the first and the second sequences. This pattern has a length of 3 because it contains three items. Another pattern is "(4 5), (3), (1)". It appears in the third and fourth sequence (it has thus a support of 0.5). It also has a length of 4 because it contains 3 items.

The PrefixSpan has following advantages:

a. No candidate generation.
b. The frequency of local items only countable.
c. Divide-and-conquer search methodology is used.
d. It is superior to GSP as well as FreeSpan.

But still there is need to improvise the PrefixSpan algorithm so as to reduce the cost for creating projected databases as well as to reduce the scanning time of projected databases.

5. RESULTS AND DISCUSSIONS

The results are drawn by executing the PrefixSpan algorithm on different datasets. Minimum support (minsup) and Maximum prefix length (MPL) are the two parameters which are specified initially, on basis of which the sequential patterns are generated. In previous research only the minimum support values are considered to get the sequential patterns through PrefixSpan algorithm. As the algorithm is tested on large datasets the additional parameter that is maximum prefix length is provided at start of execution and results are drawn based on these two parameters.

The different datasets C16D248k and C16D150k are developed using synthetic dataset generator. One more dataset C21D36k which is conversion of Bible into sequence database is also used to draw the results of algorithm execution. The C stands for average number of item sets per sequence and D stands for number of sequences in the labels of datasets.

Minimum support (MS) and maximum prefix length (MPL) values are set initially on the basis of which the sequential patterns are generated from sequential datasets. The performance of PrefixSpan algorithm on different datasets is evaluated by two parameters that are time complexity and memory utilization. The values of time complexity and memory utilization vary according to different datasets on which the algorithm is run.

The different values of minimum support (MS) and maximum prefix length (MPL) are provided initially for the execution of PrefixSpan algorithm on different datasets (C16D248k, C16D150k and C21D36k) and the results are drawn in terms of time complexity and memory utilization.

C16D248k dataset has two hundred and forty eight thousand transactions while C16D150k dataset has one hundred and fifty thousand transactions. Both of the datasets has sixteen item sets per sequence. Both the datasets are sequential datasets and are evaluated based on two parameters (minimum support and maximum prefix length) in order to find the frequent sequential patterns in time and memory efficient way. The maximum prefix length plays crucial role while executing the PrefixSpan algorithm on large datasets.

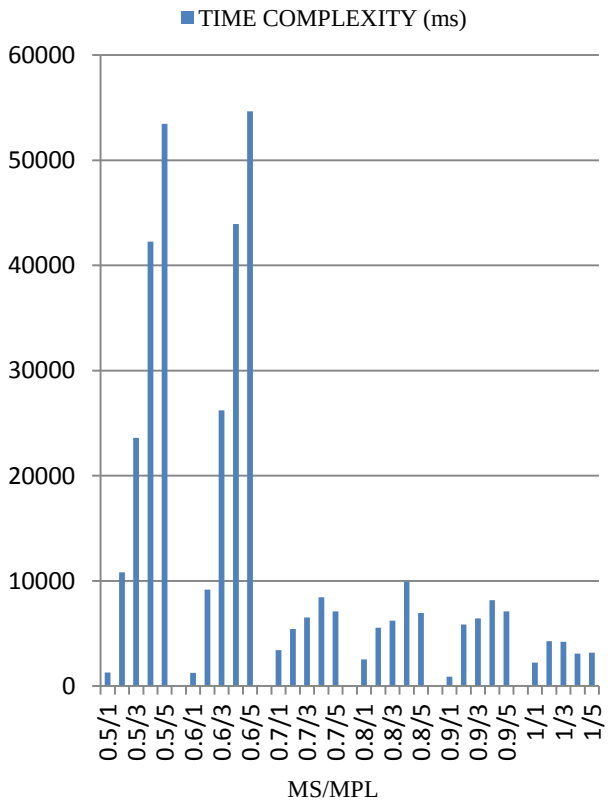

Fig 1: Time complexity Bar Graph for C16D248k

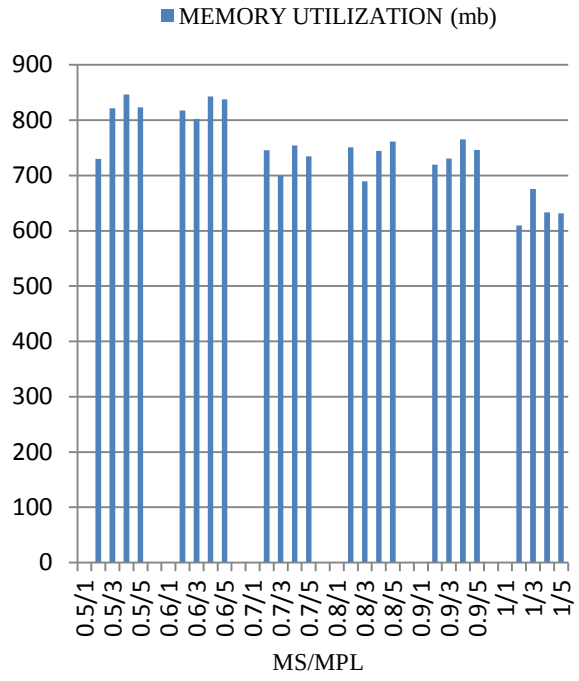

Fig 2: Memory Utilization Bar Graph for C16D248k

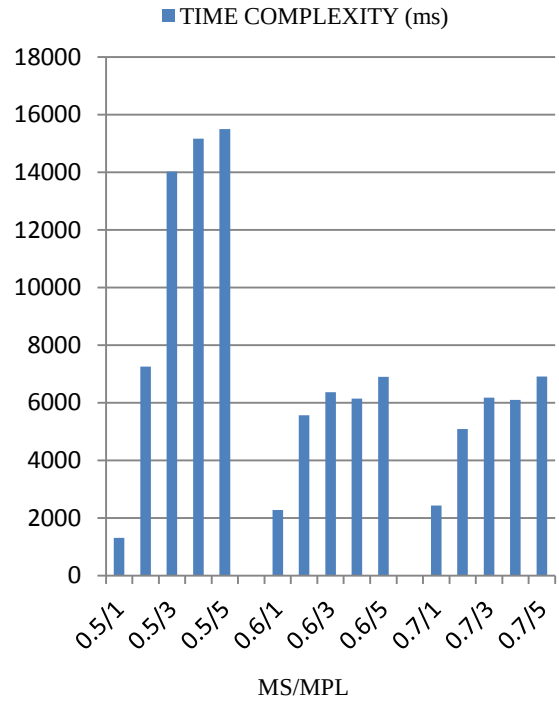

Fig 3: Time complexity Bar Graph for C16D150k

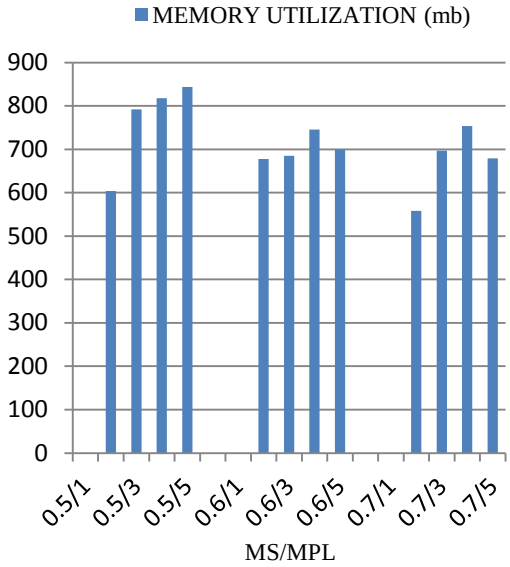

Fig 4: Memory Utilization Bar Graph for C16D150k

C21D36k dataset which is conversion of bible into sequential datasets has thirty six thousand transaction with twenty one average number of item sets per sequence. The PrefixSpan algorithm is executed on C21D36k. The minimum support and maximum prefix length values are provided at the start of the execution and these values are varied in order to evaluate the performance of algorithm. The time complexity and memory utilization is calculated based on the minimum support and maximum prefix length.

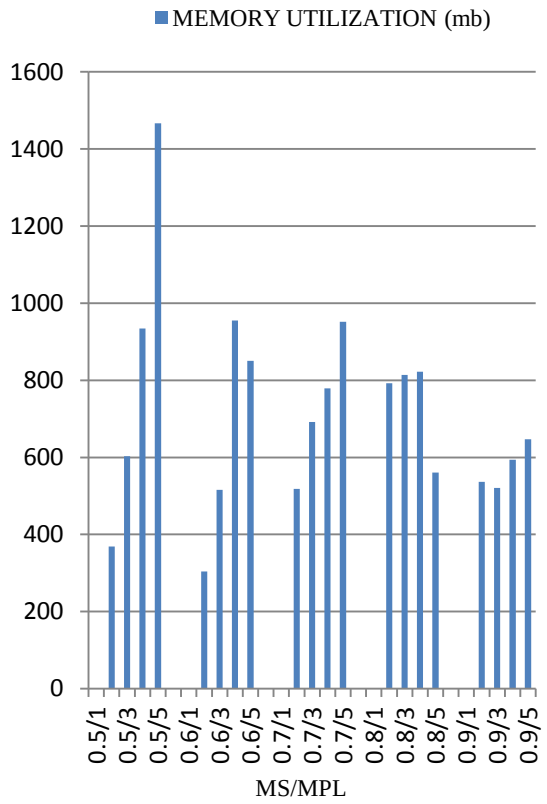

Fig 5: Memory Utilization Bar Graph for C21D36k

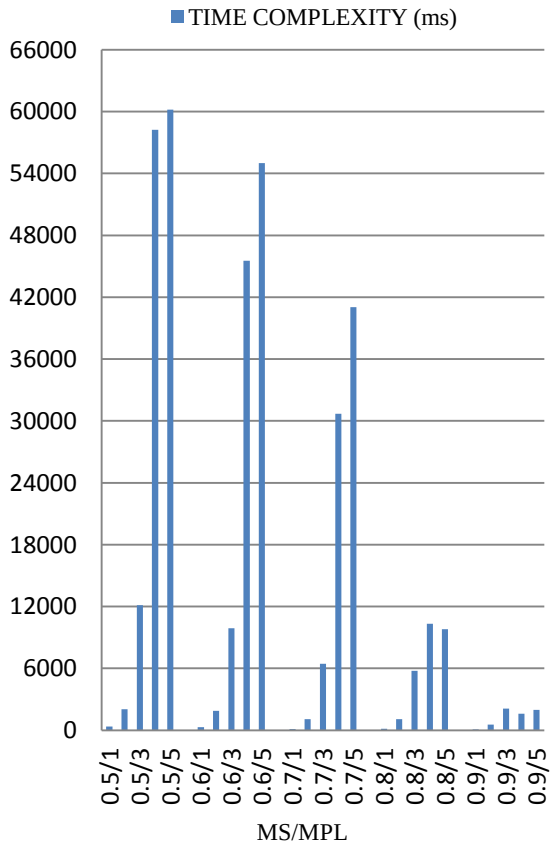

Fig 6: Time Complexity Bar Graph for C21D36k

6. CONCLUSION

The performance of existing PrefixSpan algorithm is evaluated by running the algorithm on different datasets (C16D248k, C16D150k and C21D36k). The two parameters minimum support and maximum prefix length are provided at start of the execution of algorithm. The sequences having value greater than minimum support are extracted from sequential datasets. Minimum support is the number of sequences which are calculated by dividing the pattern occurs with the total number of sequences in the database. The maximum prefix pattern value used to specify the length of the sequence to be there in output sequential patterns which is beneficial while executing the algorithm on large datasets. For getting the sequential output based on minimum support and maximum prefix length, the two parameters time complexity and memory utilization are set as the benchmark for performance evaluation of algorithm on different datasets. Both the parameters vary from one dataset to other. These results are plotted in bar graph and are useful in order to analyse the performance of existing algorithm.

The time complexity for dataset C16D248k for minimum support 0.5 and 0.6 is slightly changed but fit varies greatly from 0.7 to 1 minimum support. In case of memory utilization for C16D248, it varies slightly and goes on decreasing as minimum support increases from 0.5 to 1 and maximum prefix length increases from 1 to 5. For dataset C16D150k, the time complexity decreases as minimum support increases from 0.5 to 0.7 while the memory utilization changes slightly with change in minimum support and maximum prefix length. The time complexity for dataset C21D36k decreases significantly with change in minimum support while memory utilization is at high for 0.5 minimum support then decreases and changes slightly for 0.6 to 0.9 minimum supports.

7. FUTURE SCOPE

The improvisation can be done by applying parallel processing for large datasets and by adding some extra constraints like spacer projection and length constraints. The compressions techniques can also be used in order to reduce the projected dataset thus improve the performance of existing algorithm.

8. REFERENCES

[1] R Agrawal and R Srikant, 1995. Mining sequential patterns, In Proceedings of 1995 International Conference Data Engineering (ICDE'95), pp. 3- 14, Taipei, Taiwan.

[2] R Agrawal and R Srikant, 1994. Fast algorithms for mining association rules, In Proc. 1994 Int. Conf. Very Large Data Bases (VLDB'94), pp. 487- 499, Santiago, Chile.

[3] M. Zaki, 2001. SPADE: An Efficient Algorithm for Mining Frequent Sequences, Machine Learning, vol. 40, pp. 31- 60.

[4] Han J., Dong G., Mortazavi-Asl B., Chen Q., Dayal U., Hsu M.-C., 2000. Freespan: Frequent pattern-projected sequential pattern mining, In Proceedings 2000 Int. Conf. Knowledge Discovery and Data Mining (KDD'00), pp. 355-359. 2000.

[5] Jian Pei, Jiawei Han, Behzad Mortazavi, Umeshwar Dayal, 2004. Mining Sequential Patterns by Pattern-Growth: The PrefixSpan Approach, IEEE transactions on

knowledge and data engineering, Vol. 16, pp. 1424-1440.

[6] LIU Pei-yu, GONG Wei and JIA Xian, 2011. An Improved PrefixSpan Algorithm Research for Sequential Pattern Mining, In Proceedings 2011 International Symposium, pp. 377-380.

[7] Liang Dong and Wang Hong. 2014. A improved PrefixSpan Algorithm for Sequential Pattern Mining, In Proc. 2014 IEEE International Conference, Vol. 1, pp. 103-108.

[8] Zhou Zhao, Da Yan and Wilfred Ng. 2014. Mining Probabilistically Frequent Sequential Patterns in Large Uncertain Databases, IEEE transactions on knowledge and data engineering, Vol. 26, pp. 1171-1184.

[9] J. Pei, J. Han and W. Wang, 2007. Constraint-based sequential pattern mining: the pattern growth methods, J Intell. Inf. Syst, Vol. 28, No.2, pp. 133 –160.