IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 10, Issue 4 (Mar. - Apr. 2013), PP 12-20
www.iosrjournals.org

Sequential Pattern Mining Methods: A Snap Shot

Niti Desai[1], Amit Ganatra[2]
[1](Deprtment of Computer Engg, Uka TarsadiaUniversity,Bardoli,Surat,Gujarat, India
[2](U and P U Patel Department of Computer Engineering, Charotar University of Science and Technology, Changa 388421, Anand, Gujarat,India

Abstract: *Sequential pattern mining (SPM) is an important data mining task of discovering time-related behaviours in sequence databases. Sequential pattern mining technology has been applied in many domains, like web-log analysis, the analyses of customer purchase behaviour, process analysis of scientific experiments, medical record analysis, etc. Increased application of sequential pattern mining requires a perfect understanding of the problem and a clear identification of the advantages and disadvantages of existing algorithms. SPM algorithms are broadly categorized into two basic approaches: Apriori based and Pattern growth. Most of the sequential pattern mining methods follow the Apriori based methods, which leads to too many scanning of database and very large amount of candidate sequences generation and testing, which decrease the performance of the algorithms. Pattern growth based methods solve all above problems and in addition, it works on projected database which minimize the search space. Paper reviews the existing SPM techniques, compares various SPM techniques theoretically and practically. It highlights performance evaluation of each of the techniques. Paper also highlights limitation of conventional objective measures and focused on interestingness measures. Finally, a discussion of the current research challenges and pointed out future research direction in the field of SPM.*

Keywords: *Sequential Pattern Mining, Sequential Pattern Mining Algorithms, Apriori based mining algorithm, FP-Growth based mining algorithm*

I. Introduction:

Data mining problem, discovering sequential patterns, was introduced in [1]. The input data is a set of sequences, called *data-sequences*. Each data-sequence is a list of transactions, where each transaction is a set of literals, called *items*. Typically, there is a transaction-time associated with each transaction. A sequential pattern also consists of a list of sets of items. A sequence is *maximal* if it is not contained in any other sequence. A sequence with *k* items is called a *k*-sequence.

In addition to introducing the problem of sequential patterns, [1] presented three algorithms for solving this problem, but these algorithms do not handle following:

- Time constraints
- Sliding windows
- Taxonomies

Two of these algorithms were designed to solve only maximal sequential patterns; however, many applications require all patterns and their supports. The third algorithm, AprioriAll, find all patterns; its performance was better than or comparable to the other two algorithms which are introduced in [2]. AprioriAll is a three-phase algorithm:

Phase 1: It first finds all item- sets with minimum support (frequent itemsets)
Phase 2: transforms the database so that each transaction is replaced by the set of all frequent itemsets contained in the transaction
Phase 3: Then finds sequential patterns

There are two problems with this approach:

- It is computationally expensive to do the data transformation.
- while it is possible to extend this algorithm to handle time constraints and taxonomies, it does not appear feasible to incorporate sliding windows.

Srikant and Agrawal [10] generalized their problem to include : Time constraints, Sliding time window,User-defined taxonomy.They have presented Apriori-based improved algorithm GSP (i.e., **G**eneralized **S**equential **P**atterns). It also work on heuristic, *Any super pattern of a non frequent pattern cannot be frequent*. GSP [10], adopts a multiple-pass, candidate generation-and-test approach. SPIRIT algorithm is to use regular expressions as flexible constraint specification tool [5].For frequent pattern mining, a **F**requent **P**attern **growth** method called FP-growth [7] has been developed for efficient mining of frequent patterns without candidate generation.

FreeSpan (**Fre**qu**e**nt pattern-projected **S**equential **pa**ttern mining) [6], which reduce the efforts of candidate subsequence generation. Another and more efficient method, called PrefixSpan [8] (**Prefix**-projected **S**equential **pa**ttern mining), which offers ordered growth and reduced projected databases. To further improve the performance, a pseudo-projection technique is developed in PrefixSpan.

In the last decade, a number of algorithms and techniques have been proposed to deal with the problem of sequential pattern mining. From these, GSP and PrefixSpan are the best-known algorithms. This survey paper mainly focuses on SPM based on **A**ssociation **R**ule **M**ining (ARM). Basically there are two main methods to find the association of data items: (1) Apriori based method which is work on Generate and Test (2) Frequent pattern Growth (FP-Growth) which is Graph-based method. Both the methods are worked on frequency (minimum support).

II. Justification of Area

Data Mining is task which is finding Interesting and useful information from large data amount, which can be used in numerous areas. It can be applicable in many domains like, web-log analysis, medical record analysis, retail marketing, stock analysis, telecommunication field etc. Lot of work already been done on SPM. Environment may vary constantly. So, it is necessary to understand up-coming trend and emerging progress. Different sets of rules are used to identify sequence pattern but rules may change over a time period. So, It is necessary to indentify and incorporate novel rules in algorithm and design more efficient sequential pattern mining methods which is capable enough to identify innovative trends.

III. Related Work

3.1. Apriori based mining algorithm

The Apriori [1] [Agrawal and Srikant 1994] and AprioriAll [2] [Agrawal and Srikant 1995] worked on *"All nonempty subsets of a frequent itemset must also be frequent."* It's worked on basic strategy of Generate and Test. This follows below steps:

(i) Generate candidate
(ii) Scan DB for each candidate
(iii) Test candidate support count with minimum support count

Technique suffers from following:

(i) Repeated scanning of database
(ii) Huge sequence of candidate generation, which decreases the efficiency.

3.1.1. Apriori-based SPM Algorithms:

The sequential pattern mining problem was first proposed by Agrawal and Srikant in [1], and the same authors further developed a generalized and refined algorithm, GSP [10], based on the Apriori property [1]. Since then, many sequential pattern mining algorithms have also been proposed for performance improvements. Among those, SPADE [11], and SPAM [3] are quite interesting ones. SPADE is based on a vertical id-list format and uses a lattice-theoretic approach to decompose the original search space into smaller spaces. SPAM is a recently developed algorithm for mining long sequential patterns and adopts a vertical bitmap representation. Its performance study shows that SPAM is more efficient in mining long patterns than SPADE.
Apriori-based Methods are mainly categorized into following:

- Apriori-based, horizontal formatting method: GSP Srikant and Agrawal (1996)[10]
- Apriori-based, vertical formatting method: SPADE (Zaki, 2001) [11]
- Projection-based pattern growth method: SPAM (Ayres et al., 2002)[3].

Table 1Shows comprehensive study of existing Apriori-based algorithms.

Table 1: Comparative study of Apriori-based Algorithm

Apriori-based Algorithm			
Algorithm	**GSP** (Generalized Sequential Pattern) [10]	**SPADE** (Sequential PAttern Discovery using Equivalent Class) [11]	**SPAM (** Sequential Pattern Mining) [3]
Key features	Generate & Test	-A vertical format -Reduce the costs for computing support counts	-Improvement of SPADE - Reduce cost of merging - Lattice search techniques -Sequences are discovered in

			only three database scans
Working	Scan DB for frequent item/candidate If the candidates do not fit in memory, generates only those candidates will fit in memory. If sequence is frequent are written to disk; else removed	-Divide the candidate sequences into groups by items. -ID-List technique to reduce the costs for computing support counts.	-Represent each ID-list as a vertical bitmap -data set stored by <CID,TID,Itemsets> where,CID: customer-id, TID: transaction-id based on the transaction time
Location Memory wise	Not a main-memory algorithm	ID-List completely stored in the main memory	<CID,TID,Itemsets> Completely stored in the main memory
Data Structure	candidate sequences are stored in a hash-tree	Hash-tree (ID –list)	vertical bitmap
Limitation	-Multiple scanning -Multiple passes over the database	-Same pair is recorded more times when it/they appear(s) more than once in the same customer sequence -repeatedly merge the ID-list	(Customer id list,transaction id list and itemset) Information triplet should be in main memory.

3.2. Frequent pattern Growth (FP-Growth) based mining algorithm:

Pattern growth-method [7] is the solution method of limitations of the Apriori-based methods. It comes up with solution of the problem of generate-and-test. It's work on following key features:

1. Avoid the candidate generation step
2. Focus the search on a restricted portion of the initial database

Work on following:

1. Scan DB once, find frequent 1-itemset (single item pattern)
2. Order frequent items in frequency descending order
3. Scan DB again, construct FP-tree

It is faster than Apriori because:

- Candidate generation-testing is not performed, which uses Compact data structure
- It eliminate repeated database scanning.
- Basic operation performed is counting and FP-tree building.

3.2.1. Frequent pattern Growth based SPM algorithm:

FreeSpan [6] was developed to substantially reduce the expensive candidate generation and testing of Apriori. FreeSpan uses frequent items to recursively project the sequence database into projected databases while growing subsequence fragments in each projected database. While PrefixSpan[8] adopts a horizontal format dataset representation and mines the sequential patterns under the pattern-growth paradigm: grow a prefix pattern to get longer sequential patterns by building and scanning its projected database. PrefixSpan out performs almost all existing algorithms [8].

Table 2: Comparative study of FP-Growth based Algorithm

FP-Growth based algorithm		
Algorithm	**FreeSpan (Frequent pattern-Projected Sequential pattern mining) [6]**	**Prefixspan (Prefix-Projected Sequential Pattern Mining) [8]**
Key features	Reduce candidate generation(basic feature of FP-growth) Work on projected database	Reduce candidate generation(basic feature of FP-growth) Work on projected prefix database (Less projection)
Core idea of projection	projected sequence database on based of frequent item	Scan DB & find frequent items Recursively &Database projection based on frequent Prefix
Optimization	--	(1) Bi-level: Partition search space based on length-2 sequential patterns (2)pseudo-projection: Pointer refer pseudo-projection in sequence DB

		Projection information: pointer to the sequence in database and offset of the postfix in the sequence.
Limitation	Database projection cost	Prefix Database projection cost (lower than Database projection cost for frequent item)
advantage	Reduce search space	Projection is based only on frequent prefixes which minimize search space

IV. Experimental Results:

In this section we have performed a simulation study to compare the performances of the algorithms: Apriori [2], PrefixSpan [8] and SPAM [3], Comparison is based on runtime, frequent sequence patterns, memory utilization on various (25 % to 50%.) support threshold.

These algorithms were implemented in Sun Java language and tested on an Intel Core Duo Processor with 2GB main memory under Windows XP operating system. Dataset is generated on SPMF (Sequential Pattern Mining Framework) software. Following is the description of Dataset:

Table 3: Description of Dataset

Number of distinct items	100
Average number of itemsets per sequence	7.0
Average number of distinct item per sequence	29.5
Average number of occurences in a sequence for each item appearing in a sequence	1.18644
Average number of items per itemset	5.0

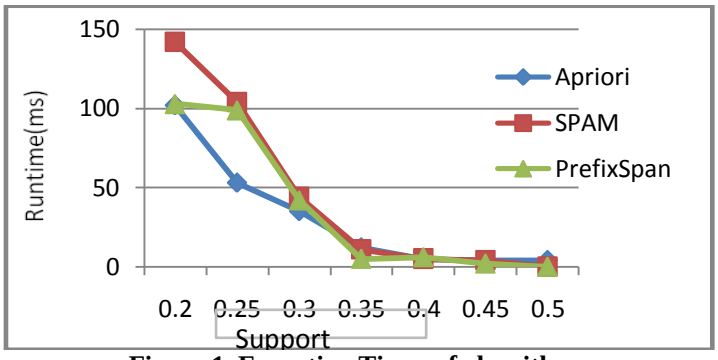

Figure 1. Execution Times of algorithms

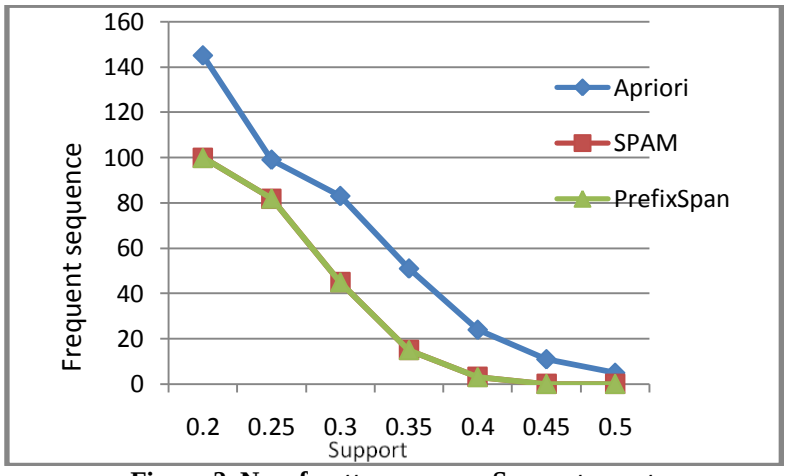

Figure 2. No. of patterns verses Support count

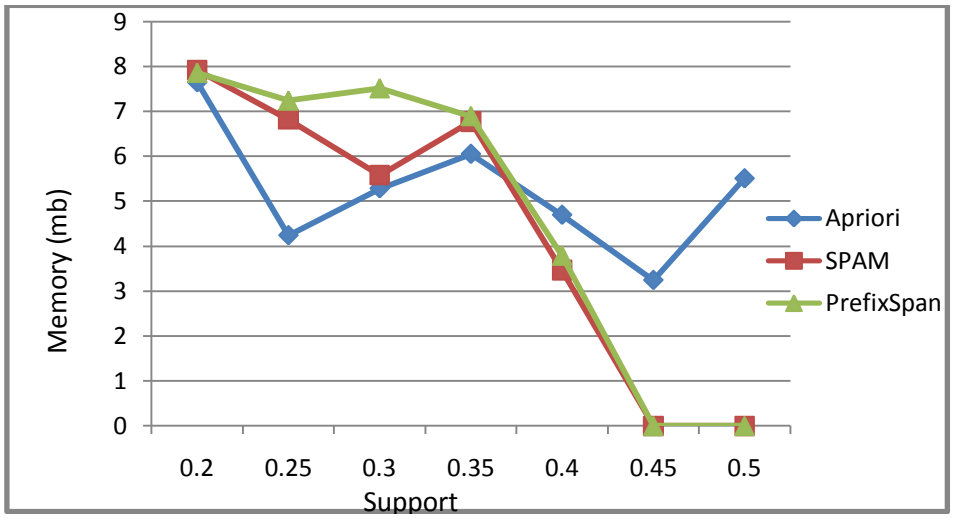

Figure 3. Memory utilization of algorithm

On comparing the different algorithms above results have been obtained. The following points can be observed from above simulation:

- Time taken for lower support is almost half to double for SPAM and PrefixSpan as compare to Apriori. Gradually time taken by SPAM and PrefixSpan are decreased as compare to Apriori. SPAM and Apriori taking same time to execute in case of support range 0.30-0.45.Same PrefixSpan has taken less time in a same state.
- Same no. of frequent sequence are generated with SPAM and PrefixSpan which are less than Apriori.
- For a lower support memory consumption is less in case of Apriori but for medium support range memory consumption is reduced by 49% in SPAM and 45% in PrefixSpan.
- In all above cases SPAM and PrefixSpan drawn good results but PerfixSpan really perform better in case of execution time of algorithm.

Above discussed SPM algorithms worked on objective measures: (i) support (ii) confidence

Support: The Support of an itemset expresses how often the itemset appears in a single transaction in the database i.e. the support of an item is the percentage of transaction in which that items occurs.

Formula: $\mathbf{I = P(X \cap Y) = \frac{(X \cap Y)}{N}}$

Range: [0, 1]
If I=1 then Most Interesting
If I=0 then Least Interesting

Confidence: Confidence or strength for an association rule is the ratio of the number of transaction that contain both antecedent and consequent to the number of transaction that contain only antecedent.

Formula: $\mathbf{I = P\left(\frac{Y}{X}\right) = \frac{P(X \cap Y)}{P(X)}}$

Range: [0, 1]
If I=1 then Most Interesting
If I=0 then Least Interesting

Comment on existing objective Measures:

- Support is use to eliminate uninteresting rule. Support indicates the significance of a rule, any rules with very low support values are uncommon, and probably represent outliers or very small numbers of transactions but sometimes low value support data is interesting or profitable.
- Confidence measures reliability of the inference made by the rules. Rules with high confidence values are more predominant in the total number of transactions. We can also say that confidence is an estimate of the conditional probability of a particular item appearing with another.

A rule (pattern) is interesting if (1) Unexpected: pattern which is extracted is surprising to the user (2) Actionable: user can utilize resultant pattern further.

Several interestingness measures for association rule is recommended by Brijs et al., 2003 [4]. Ramaswamy et al. developed the objective concepts of *lift* to determine the importance of each association rule [9]. Here in this paper we have chosen an improvement in the "%Reduction". % Reduction denotes the percentage of rules discarded. It is denoted by below formula:

% Reduction= (No. of rules rejected / No. of rules on which mining was applied) *100

Lift: It is a measure which predicts or classifies the performance of an association rule in order to enhance response. It helps to overcome the disadvantage of confidence by taking baseline frequency in account. [4] [9]

Formula: $\mathbf{I = \frac{P(X \cap Y)}{P(X) * P(Y)}}$

Range: [0, ∞]

If 0 < I < 1 then X□ Y are negatively interdependent If I=1 then interdependent If ∞ > I > 1 then X□ Y are positively interdependent

We have done experiment on interestingness measure *lift*. And compare with existing measures support and confidence.

Table 4: Sample Dataset 2

REGION	HAIR	GENDER	WORK	HEIGHT
West	Brown hair	Female	Stitching	Tall
West	Black hair	Female	Cooking	Tall
West	Black hair	Male	Painting	Medium

Table 5: Association Rules generated after applying Apriori algorithm on Sample dataset2 (Table 4)

Antecedent □ Consequent □
{West, emale} □ {Tall} □
{West, Tall} □ {Female} □
{ Female, all} □ {West} □
{Tall} □ {West, Female} □
{Female} □ {West, Tall} □
{West} □ { Female, Tall} □

Table 6: The comparison of Interestingness values for all measures

Association Rule	Support	Confidence	Lift
{West, Female} □ □{Tall}	0.666667	1	1.5
{West, Tall} □ □{Female}	0.666667	1	1.5
{ Female, Tall} □{West}	0.666667	1	1
{Tall} □ □{West, Female}	0.666667	1	1.5
{Female} □ □{West, Tall}	0.666667	1	1.5
{West} □ □{ Female, Tall}	0.666667	0.666667	1

Table 7: % Reduction values for all Measures

Interestingness Measures	% Reduction
Support	0
Confidence	0
Lift	33.33

We observed following: Lift gives a high % Reduction in the sample dataset above. Conventional measures Support and Confidence gives poor % reduction i.e. zero.

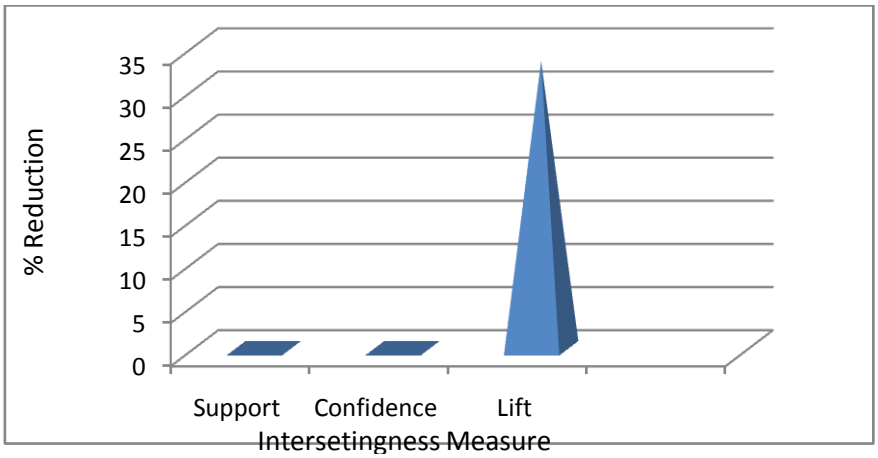

Fig 4: % Reduction values for all Measures

Therefore, a comparative study was drawn on the three measures of Support, Confidence and Lift on two more small sample datasets and one large dataset (ref. Table3). We have taken standard values of Support and Confidence for dataset 2 and dataset 3 taken to carry out the comparison: Support=30%, Confidence=30%.

Table 8: Sample dataset 2

TID	**ITEMS BOUGHT**
ID1	Scale, Pencil, Book
ID2	Pen. Pencil, Rubber
ID3	Scale, Pen, Pencil, Rubber
ID4	Pen, Rubber

Table 9: Sample dataset 3

TID	**ITEMS BOUGHT**
T1	Bread,Butter,Milk,Beer,Sandwich
T2	Bread,Butter,Milk
T3	Milk,Bread,Jam,Sandwich,Beer
T4	Beer,Jam,Curd,Sandwich

Table 10: Comparison of % Reduction between Support, Confidence and Lift

Dataset / **Measure**		**Dataset 1**	**Dataset 2**	**Dataset 3**
Support	0.2	0	0	0
	0.4	100	0	0
	0.6	100	100	100
	0.8	100	100	100
Confidence	0.2	0	0	0
	0.4	3.33	0	0
	0.6	21.11	0	0
	0.8	21.11	42.85	66.667
Lift		6.667	14.28	33.33

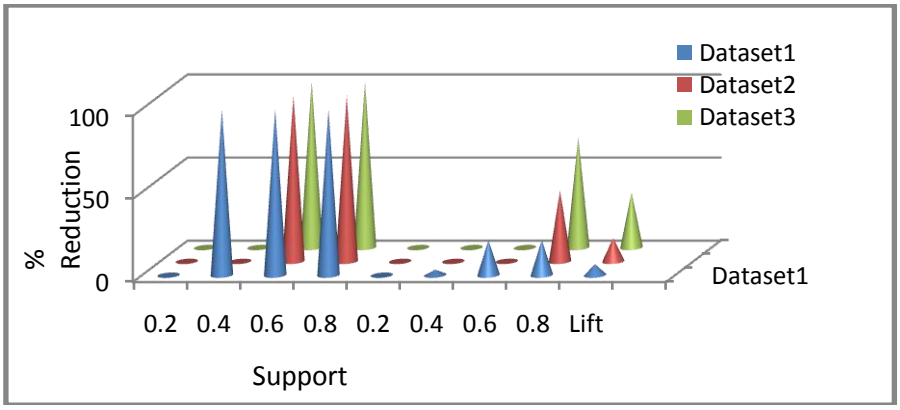

Fig 5: % Reduction values for all Measures

Following comparative study was drawn for large database described in table 3. We have used conventional FP-Growth (without lift) and FP-Growth with lift algorithms where we have kept confidence is 0.30.We simulated experiment for three measures: Support, Confidence and Lift.

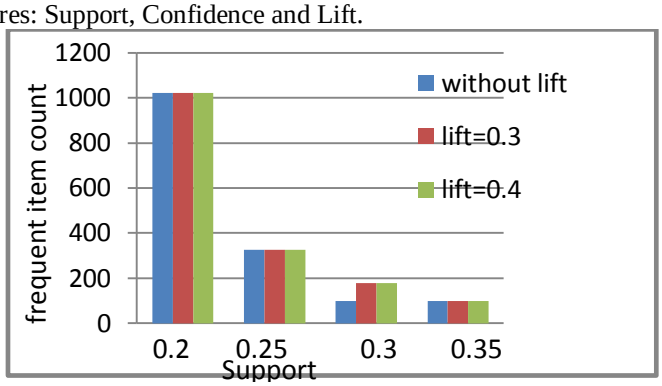

Fig 6: Frequent item count generated by FP-growth with lift and without lift

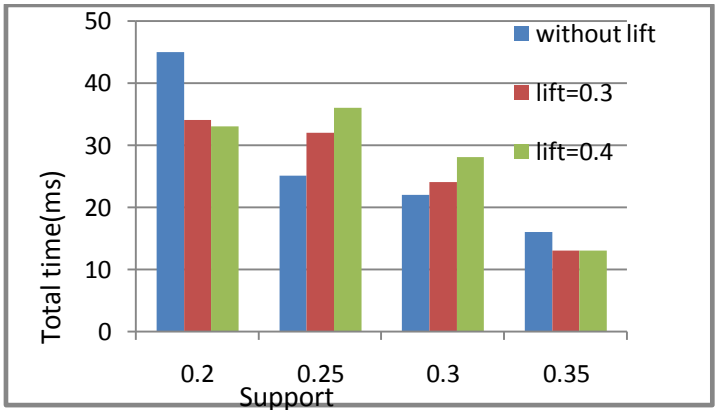

Fig 7: Execution time of FP-growth with lift and without lift

Table 11: Association rules generated for Dataset 1 (ref. Table 3)

Association rule generation states (confidence=0.30)			
Support	**lift**	**No .of association rule generated**	**Time(ms)**
0.20	**--**	**4547**	**151**
	0.30	**112**	**62**
	0.40	**112**	**63**
0.25	**--**	**727**	**21**
	0.30	**6**	**9**
	0.40	**6**	**11**
0.30	**--**	**238**	**6**
	0.30	**2**	**4**
	0.40	**2**	**6**
0.35	**--**	**98**	**6**
	0.30	**0**	**2**
	0.40	**0**	**2**

On comparing the three measures of Support, Confidence and Lift on the basis of % Reduction, the following results have been obtained. The following points can be observed from the above Experiments:

i. Users can select their measure of interest as per their business needs with different support-confidence threshold values given.
ii. Lift gives a high % Reduction as compare to Support and confidence (fig.4 and fig.5) As per the need and rule Lift can be selected.
iii. A % Reduction of 100 is not suitable, as it indicates an exclusion of all rules, leaving no rules considered, hence the purpose of selection of actionable rules is defeated.

iv. Almost same no. of frequent sequence count is generated in both FP-Growth with lift and FP-Growth without lift.(fig.6)
v. Time taken to generate association rule in case of FP-Growth with lift is 52%-66% lower than FP-Growth without lift because almost 96%-99% less rules are generated in case of FP-Growth with lift. In case of support ≥ 35 is not generated any association rule which is not favourable to lead to actionable rules.(fig 7)
vi. Lift worked better then confidence and support in terms of generation of association rules and time taken to find associations.(Table 11)

V. Conclusion and Future Scope:

From the theoretical and simulation study of various sequential pattern mining algorithms, we can say that PrefixSpan [8] is an efficient pattern growth method because it outperforms GSP [10], FreeSpan [6] and SPADE [11]. It is clear that PrefixSpan Algorithm is more efficient with respect to running time, space utilization and scalability then Apriori based algorithms. Most of the existing SPM algorithms work on objective measures Support and Confidence. Experiments shows % Reduction of rule generation is high in case of interestingness measures lift. Use of interestingness measures can lead to make the pattern more interesting and can lead to indentify emerging patterns.

SPM is still an active research area with many unsolved challenges. Much more remains to be discovered in this young research field, regarding general concepts, techniques, and applications.

- Researchers can identify novel measure which can make the pattern more interesting and can be helpful to identify emerging patterns.
- Research can make in such a direction where algorithm should handle large search space by modification of existing algorithm or designing novel approach.
- Algorithm should avoid repeated scanning of database during mining process which can improve efficiency of algorithm.
- To design such a SPM algorithm, which can be efficiently perform in distributed/parallel environment.

References:

[1] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," Proc. 1994 Int'l Conf. Very Large Data Bases (VLDB '94), pp. 487-499, Sept. 1994.
[2] Agrawal R. And Srikant R. 'Mining Sequential Patterns.', In Proc. of the 11th Int'lConference on Data Engineering, Taipei, Taiwan, March 1995 [3]AYRES, J., FLANNICK, J., GEHRKE, J., AND YIU, T., 'Sequential pattern mining using a bitmap representation', In Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining-2002.
[4] Brijs, T., Vanhoof, K. and Wets, G. (2003), 'Defining interestingness for association rules', *International Journal of Information Theories and Applications* 10(4), 370–376.
[5] M. Garofalakis, R. Rastogi, and K. Shim, 'SPIRIT: Sequential pattern mining with regular expression constraints', VLDB'99, 1999.
[6] Han J., Dong G., Mortazavi-Asl B., Chen Q., Dayal U., Hsu M.-C.,' Freespan: Frequent pattern-projected sequential pattern mining', Proceedings 2000 Int. Conf. Knowledge Discovery and Data Mining (KDD'00), 2000, pp. 355-359.
[7] J. Han, J. Pei, and Y. Yin, 'Mining Frequent Patterns without Candidate Generation',Proc. 2000 ACM-SIGMOD Int'l Conf. Management of Data (SIGMOD '00), pp. 1-12, May 2000.
[8] J. Pei, J. Han, B. Mortazavi-Asi, H. Pino, 'PrefixSpan: Mining Sequential Patterns Efficiently by Prefix- Projected Pattern Growth', ICDE'01, 2001.
[9] Ramaswamy, S., Mahajan, S. and Silberschatz, A. (1998), On the discovery of interesting patterns in association rules, *in* 'Proceedings of the 24rd International Conference on Very Large Data Bases',Morgan Kaufmann Publishers Inc., pp. 368–379.
[10] Srikant R. and Agrawal R.,'Mining sequential patterns: Generalizations and performance improvements', Proceedings of the 5th International Conference Extending Database Technology, 1996, 1057, 3-17.
[11] M. Zaki, 'SPADE: An Efficient Algorithm for Mining Frequent Sequences', Machine Learning, vol. 40, pp. 31-60, 2001.