CHAPTER 5

USER SPAN PATTERN FOR FUTURE PREDICTION

5.1 INTRODUCTION

Most of the web sites are interlinked pages with highly complicated structure. Thus when data is mined from usage logs it can be highly unstructured. Users of a site may have different browsing patterns over time. For predicting future behaviour of user in a site similarity in usage pattern is studied. Number of times a user has visited the same category is taken from usage logs in different sessions of equal interval. Segmentation of the log data is done to select the desired information only. Then we follow a clustering approach for selecting the matching patterns for suggesting the next possible visit of user.

Among the many error factors to be considered in usage pattern possibility of occurrence of uncertain sequences is remarkable. In the sequential pattern matching algorithm Prefix Span, Jian Pei (2004), a pattern growth approach is adopted. For better personalization in addition to prefix, a user based scan is performed in our new USP(user span pattern) algorithm. We had first sorted the data and when the count of repeated patterns for predicting possibility of future visit is considered, a threshold based approach rather than time based one is adopted.

The organization of this chapter is as follows: In section 5.2, the motivation behind this idea is briefed. In section 5.3, an outline of the algorithms used for sequential pattern analysis is explained. Related algorithms studied leading to proposed method is explained in subsections.
Section 5.4 briefs the proposed study for sequence prediction of future visit and automatic personalization. Comparison of our methodology with related study is given in Section 5.5. Section 5.6 gives a conclusion of the chapter.

5.2 MOTIVATION

Finding the navigation pattern of users, grouping the users with similar pattern and finally to predict the users’ future visit based on these patterns is a greater challenge. When considering the patterns we need to consider other factors which may directly or indirectly influence pattern mining. Uncertain patterns, colossal patterns which are long sequences of patterns and may contain many sub patterns need to be considered. Mostly studies [Agrawal & Srikanth (1995), Mannila et al. (1997)] focused on Apriori like approach for finding the sequential occurrence of patterns from web logs. Agrawal & Srikanth (1995) suggested two algorithms Apriori All and Apriori Some for pattern selection. The main difficulty with apriori like approach was the long sequence of candidate generation and is found to be a time consuming process.

Bamshad Mobasher et al. (2000) proposed a customized system based on user preferences. Here the recommendations are purely based on history alone. The system does not try to extract sequence faster by considering the possibility of partially ordered sequences due to the regular pattern of use.

Jian Pei et al. (2004) incorporated a pattern growth method in Apriori algorithms for better pattern sequence generation. Even though it is much faster than Apriori, the method suffers from the problem of growing FP tree. It is easy to read the frequent item sets once the tree is made but it is also a time consuming process. Prefix span is found to be efficient for large data
sets but suffered from the difficulty of the FP tree not fitting in to available memory.

Another remarkable challenge is possibility of occurrence uncertain sequences. Calders et al. (2010) studied the possibility of uncertain data based on statistical methods. They introduced precision and recall to ensure accuracy over the possibility of real support being under estimated or over estimated.

Yuxuan Li et al. (2013) proposed a novel approach for extracting sequences from uncertain databases. They considered gap constraints when frequent patterns are mined from large uncertain databases. They followed a dynamic programming approach for sequence finding by assuming close spatial proximity of objects. Objects may not be together even if they have close proximity when locations are considered. Uncertainty in objects identity is still a challenge.

Ours is an attempt to predict the future visit incorporating all these factors. For pattern study we selected educational site as case. In virtual learning environments we observed many pattern combinations. On close analysis of VL Schools we observed

1) Users may learn topics from introductory to advanced level.

2) Some users with basic knowledge of subject may directly start from medium level.

3) Some may not follow any pattern at all

4) Some may skip some topics which they may feel uninteresting. They may visit those pages later. That is there may be possibility of uncertain patterns.
We observed a direct relationship between the weight on a particular sequence and the number of cooccurrences of the sequence. We recorded the backward traversals also in the projected database. Based on all these we were able to come up with a better prediction model which give better success rates of prediction.

5.3 OUTLINE OF THE ALGORITHMS FOR SEQUENTIAL PATTERN MINING

In this section we present few algorithms in sequential pattern mining considered for progress in study. We present core apriori like algorithms , BFS and DFS pattern search based algorithms and pattern growth based algorithms which were closely studied to arrive at our proposal.

5.3.1 AprioriAll

Agarwal & Srikant (1995), proposed modified apriori termed as AprioriAll, which improves the performance of apriori. It improves performance by a pruning and a difference in candidate sequence generation. The algorithm traverses through five phases to get the final sequential pattern. The phases are

- Sort Phase: sorts the database on customer id based on transaction time to produce a set of sorted customer sequences.

- LargeItemset Phase: commonly known as Litemset, a scan of the present database is performed to get large 1-itemsets.

- Transformation Phase: the 1-sequences obtained is transformed to a new customer sequence with the generated Litemset. Mapping of item sets to integer values even with minimum support of Litemset is done in this phase.
• Sequence Phase: The time consuming candidate sequence generation is done during this phase. After a number of passes a sequential pattern is generated during this phase.

• Maximal Phase: Extraction of the maximal sequence from the sequential pattern is done during this last step. Non maximal sequences are pruned during this phase to produce the optimal candidate sequence.

AprioriAll produces better results than association rule mining as it takes the time order into account. But it has the major demerit of the time consuming large number of passes to produce the final sequential pattern.

AprioriSome is another sequential pattern mining algorithm by the same authors. This algorithm cares about the maximality support while the AprioriSome cares the minimality more. From the generated sequence AprioriSome selects only those ones which have maximal support. When marked in terms speed of operation AprioriSome gets sequential patterns faster.

5.3.2 BFS Method Based Algorithms

Apriori algorithms can be constructed based on the item sets that are represented in a tree like format. Many algorithms obtain sequences from iterations by traversing through the tree either in Breadth First Search or Depth First Search manner. Few of the algorithms following BFS traversal strategy for generating sequences are listed below.

5.3.2.1 Generalized Sequential Patterns (GSP)

Proposed by Srikanth & Agarwal (1996) with base of Apriori, GSP, adds time constraints to achieve efficient mined patterns. It incorporates
the idea of sliding window, with size as difference in maximal and minimal transaction time, to generate better sequences. GSP operates through two phases namely forward and backward. Both the phases it follows a gap centric approach. GSP finds the generalized patterns by working over different layers based on users style. When compared with AprioriAll and AprioriSome, GSP generates lesser candidate sequence and produces better frequent sequential patterns. But it has the limitation of performance degradation in larger data sets.

5.3.2.2 MFS and MFS+

MFS is an idea put forward by Zhang et al. (2001), follows a successive refinement approach to obtain the sequential patterns. The sequences obtained in each previous mining is taken as the estimate for proceeding till it get the final sequence. The generation method MGEN produces variable length sequences from a frequent sequence. It is a BFS based algorithm and is an enhancement of GSP. It effectively reduces the I/O cost of GSP by generating only the needed candidates for the next sequence. MFS+ proposed by the same authors selects the likely to occur sequential pattern from mined sequence found in database at a specified time and sequence obtained from updated database. This algorithm still reduces the number of iterations than MFS as it produces long sequence in early stages itself. I/O cost reduction is more in this algorithm in comparison with GSP and MFS.

5.3.3 DFS Method Based Algorithms

Depth First Search method based algorithms consumes a lot of memory space as a result of the enormous number of candidate sequences generated during the sub graph generation stages. Such algorithms may also
exhibit negative pruning result. Few algorithms following DFS method is listed below:

5.3.3.1 Sequential Pattern Discovery Using Equivalence Classes

Sequential PAttern Discovery using E quivalence classes (SPADE) proposed by Mohammed Zaki (2001) is an efficient DFS based algorithm. It uses a vertical ID list and associates sequences with generated timestamp. It generates Temporary ID list for all the candidates and represents it in a lattice space. It uses temporal joins to generate patterns from the temporary ID list. The support for a k-sequence can be obtained by looking at any two of the (k-1) subsequences intersection. SPADE is very economic in terms of computational cost and is having reduced number of database scans when compared with GSP. It has linear scalability based on the number of candidates and database parameters.

5.3.3.2 Frequent pattern projected sequential pattern mining

Frequent pattern projected sequential pattern mining –freespan method, by Jiawei Han et al. (2000), proposes the idea of combining the mined frequent sequences and the frequent patterns to form a projected sequence database. Follows a pattern growth method and it generates a reduced candidate subsequence. The algorithm first constructs a frequent item representation, generates length-2 sequential patterns, generates item repeating patterns and finally does recursive mining on projected database until all the patterns are obtained. In freespan the subsequence mining is limited to the projected database and thus it produces sequences faster than GSP. Less candidate sequences are generated in freespan as it generates sequence based on projected database.
5.3.3.3 PrefixSpan

Jian Pei et al. (2004) proposed a novel sequential pattern mining method, PrefixSpan, which follows a prefix projected pattern growth method for generating sequences. Patterns grow from smaller to larger by concentrating on subspace. Subspaces are divisions of the search space. The algorithm progresses based on the projected database and do not need candidate generation. Prefixspan algorithm starts by scanning the database to produce frequent large 1 patterns. The database space is then partitioned based on the number of large1 item sets. Then for each item as prefix find patterns and produce length 2 sequence. Recursively all the iterations are done till there is no more element to process. Prefix span operates on a divide and conquer strategy by dividing the subsets until the pattern is obtained. During the operation the projected database is shrinks in size. It does not contain all the generated candidates but only suffix of subsequences.

The input given to prefixspan is the database and the minimum support. Output will be the list of sequential patterns.

The algorithms works in the following sequence:

- Find all the length-1 sequences for the 5 items
  \(<a>, <b>, <c>, <d>, <e>\)

- We divide the sequential patterns to number of items subsets, ie patterns with prefix \(<a>\) to patterns with prefix \(<e>\)

- Length-2 patterns are then found with each prefix. Proceed till the database is empty.
Figure 5.1 shows the sequence generation of prefixspan algorithm for a sample sequence.

Prefixspan suffer from the serious issue of cost of projected database which is to some extend overcome by the bi level projection matrix. Prefix span mines pattern more effectively than other methods but it does not consider the partially ordered sequences that may be possible in patterns. It does not consider the probability factor of such occurrences. Improved sequence finding and thus better future visit prediction can be achieved by considering these factors.
5.3.3.4 Pre order linked wap tree mining

Yi Lu & Ezeife (2005) came up with the idea of assigning position codes to the WAP Tree to avoid intermediate tree generation phases in pattern sequence extraction. The idea known as PLWAP Mine (Pre Order Linked WAP Tree Mining) does an effective attaching of binary position codes to the tree. By comparison of these position codes suffix trees of any frequent sequence under consideration is obtained. This technique effectively avoids recursive reconstruction of intermediate trees.

Position codes are assigned to a node based on some rules.

Root is assigned a position code of Null.

The leftmost node from root is assigned with a code 1

If any node is a left most child, it’s code is generated by appending 1 to the position code of parent. 10 will be added if the node is the second leftmost child.

For third leftmost child 100 will be added and the procedure is repeated.

In general position code for the $i^{th}$ leftmost child is obtained by appending the binary number $2^{n-1}$ to the parent’s code.
Figure 5.2 Position codes assigned and corresponding node positions in tree

Identification of suffixes and traversal of the PLWAP Tree to find the prefix pattern sequences is done in the pre order fashion. After finding the support of events by scanning database, frequent categories with support greater than or equal to minimum support are selected. The prefix tree structure commonly known as PLWAP Tree is then constructed by deleting the non frequent categories from the sequence. The algorithm proceeds in pre order fashion using prefix conditional sequence pattern until all the suffix
trees are traversed. Generating process of PLWAP Tree by pre order traversal is shown in Figure 5.3.

Figure 5.3 PLWAP Tree generation by pre order traversal
Peiyi Tang et al. (2007) proposed a modification to the PLWAP tree by finding the first occurrences by traversal of a portion of the WAP tree. The tree is named as First Occurrence Linked WAP Tree (FLWAP Tree). The patterns could be extracted faster in this method. Like the PLWAP algorithms once the FLWAP Tree is generated position codes are attached to nodes to find the first occurrences.

5.4 PROPOSED METHOD OF SEQUENCE GENERATION

For predicting future behaviour of user in a site similarity in usage pattern is studied. His/her browsing patterns may vary over time. In chapter 2, we proposed a probability based and weight centric method for classifying usage data. Here we elaborate more to find the sequence/pattern of traversal.

5.4.1 Future Sequence Recommendation

Prefix span pattern effectively generates patterns by pattern growth method but is not user centric. The algorithm is not for personalizing users’ requirement it only does the pattern extraction. With other algorithms candidate sequence generation is found to be a big problem.

In the category database CD , let

\[ S = \{ u_1, u_2, u_3, \ldots, u_n \} \]

is the set of users, \( \{ A, B, C, \ldots \} \) is the list of categories and

\[ \forall a \in A \quad \text{for } i: 1 \ldots n \quad a_i \subseteq A \; ; \]

\[ \forall b \in B \quad b_i \subseteq B \quad \text{and so on.} \]

Similarly

\[ \text{for } j=1 \ldots n \quad a_{ij} \ldots \subseteq Aa_i \; ; \quad b_{ij} \ldots \subseteq Bb_i \quad \text{and so on.} \]

The proposed algorithm UserSpanPatten find sequence and predict future behaviour of user based on the following assumptions:
• There can be relationship between categories and their possibility of occurring together in sequences. This could be extracted by studying frequent patterns generated from user logs. The relationship can be based on Association rule of Rakesh Agarwal (1993). This relationship can contribute to sequence recommendation and thus personalization.

Let \( C = \{C_1, C_2, C_3, \ldots, C_m\} \) denote \( m \) frequent categories sets, that is the groups of categories having a possibility of co-occurrence.

and \( T = \{t_1, t_2, t_3, \ldots, t_p\} \) set of \( p \) transactions and \( \forall t \in T \), \( C_k \subseteq t \)

Support for category \( C_k \) is the number of transactions in \( C_k \)

\[
\text{Sup}(C_k) = \frac{|C_k \subseteq t|}{|T|} \quad (5.1)
\]

As per association rule, support of a rule \( C \Rightarrow D \) is \( \text{Sup}(C \cup D)/|T| \) and confidence is \( \text{Sup}(C \cup D)/\text{Sup}(C) \). Association rule can easily find relationship in categories based on the patterns in transactions. If any item fails in minimum support such items can be pruned as there is no possibility of such sequences to occur. Based on this we can reduce the search space.

• There may be uncertain occurrences of sequences. Possibilities of finding certainty from uncertainty will be a good solution to overcome this problem.

A detailed scan of the weight tree, and based on the existing pattern structure probability of occurrence of frequent patterns can be obtained. Uncertain occurrences can also be extracted. An FP tree like structure can be used to represent the sequence. The uncertainty factor denoted as \( rU \) is found to be minute but it increases as level increases.
- Possibility of partially ordered sequences in patterns to be considered.

For the sample database

**Table 5.1 Sample sequence database for user1**

<table>
<thead>
<tr>
<th>User</th>
<th>Category</th>
<th>Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>u1</td>
<td>A</td>
<td>{Aa₁a₂, A a₁ a₂ a₁₂a₂ a₃, A a₂ a₁₂ a₃ a₃₁a₄a₁, A a₁ a₁₂ a₃ a₃₁ a₃₁ }</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>B b₁ b₃, B b₁b₁₁ b₂ b₃, B b₁ b₂ b₁ b₃</td>
</tr>
</tbody>
</table>

The algorithm works as follows:

For particular session Se

Step1: Find the total count of category. The category with highest count is selected. This step is already done based on weight in the classification stage. The first personalized category in the classification stage will be set as the most probably to be selected category.

Now our sequence to process is

**Table 5.2 Length-1 sequence table based on user preference**

<table>
<thead>
<tr>
<th>User</th>
<th>Category</th>
<th>Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>u1</td>
<td>A</td>
<td>{Aa₁a₂, }</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{ A a₁ a₂ a₁₂a₂ a₃ }</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{ A a₂ a₁₂ a₃ a₃₁a₄a₁ }</td>
</tr>
<tr>
<td></td>
<td></td>
<td>{ A a₁ a₁₂ a₃ a₃₁ a₃₁ }</td>
</tr>
</tbody>
</table>
Step 2: Scan for repeated patterns in the current database

Find the total count of subcategories from subsequence as \( a_1: 5, a_2: 4, a_3: 3, a_4: 1 \) and \( a_{12}: 2, a_{21}: 1, a_{31}: 3 \)

Based on support and count \( a_1, a_2, a_3 \) and \( a_{12} \) and \( a_{31} \) can be considered.

But since we consider partially order sequence for frequent pattern, next is to scan for ordered sequences.

If we have higher levels/order of traversals we need to consider that also during this stage.

Step 3: Consider the ordered patterns

Count of

\[
\begin{align*}
\text{Count of} & \quad a_1a_{12} : 2 \\
& \quad a_3a_{31} : 2 \\
& \quad a_2a_{21} : 1
\end{align*}
\]

Considering ordered sequence and total number of count, \( a_1 : 5 > a_3 : 3 \). Select \( a_1 \) and the sequence \( a_1a_{12} \) has higher possibility of occurrence. The most possibly to occur pattern is \( \{ Aa_1a_{12} \} \). This is set as recommended sequence of future visit for the particular user. This recommendation is also a way to achieve faster traversal.

The sequence ordering is also a matter of consideration. Pattern \( a_1a_{12} \) and \( a_{12}a_1 \) are considered different. This assumption is based on the fact that the order change may affect in link weight calculation factor. As an additional measure to ensure accuracy prediction of future visit pattern, the traversal patterns in different sessions of equal duration are considered. This order sequence consideration, weight based category sorting and session consideration in our algorithm helps in extracting exact interest of user.
5.4.2 Clustering

The sequence database is segmented to form clusters of similar sequences. Clustering was done based on the following considerations.

- Similarity of users is studied.
- The clustering is based on implicit frequent patterns obtained by scan of the sequence database. The weight factor of clusters recommends the possibility of frequent sequence. As observed the direct relationship of weight on a page and frequent sequence can also enhance user personalization.
- We selected fixed size active sessions, may be termed as windows, for finding match between clusters.

Users of similar browsing patterns obtained from this cluster information can be effectively used to personalize the system. In virtual learning environment we can provide selected courses to this user group and can achieve effective learning.

For each cluster $c_i \in C_l$, the set of clusters and

for sessions $Se = \{se_1, se_2, se_3, \ldots se_n\}$ at time $t$

Match possibility between clusters in session $se$

$$MSc(se, cl) \propto wfp(se, cl)$$  \hspace{1cm} (5.2)

Where $wfp(se, cl)$ is the weight of frequent pattern of cluster $cl$ in session $se$ and is extracted from querying the database. Also $se \neq \Phi$.

5.5 COMPARISON WITH RELATED STUDY

This section draws the similarity and differences of the proposed method with core sequence generation algorithms. Comparison with Apriori
like algorithms, rule growth and pattern growth algorithms are done. UserSpanPattern follows a user centric pattern growth approach.

5.5.1 Comparison with Apriori Like Algorithms, GSP and SPADE

- AprioriLike algorithms follow association rule to draw relationship between categories and their possibility of occurring together in sequences. The proposed algorithm also checks the probability based on association rule.

- The main drawback of these algorithms are the long sequence of candidate generation during successive scans of the database. GSP generates comparatively lesser candidate sequence but it fails for larger database. On the other hand this study does not generate large candidate sequence. It works from the reduced database after each scan.

- SPADE based on the DFS method of traversal is very economic in terms of computational cost and is having reduced number of database scans. It has linear scalability based on the number of candidates and database parameters. The proposed method follows a tree structure which is similar to SPADE. But the traversal is not DFS pattern. It first selects the top listed category, makes it as the first node. Then we need to do weight tree traversal only for the most frequent occurred category. Recommendations are done for personalizing user traversals. This reduced traversals makes the algorithm to converge faster.
5.5.2 Comparison with FreeSpan and PrefixSpan

FreeSpan is an efficient method for frequent items mining and is found to perform better than GSP algorithm. The proposed method also does frequent category selection projections are based on initial database which may lead to expensive implementation. The proposed method finds the frequent occurrences based on the weight and count of similar items from the reduced database each time. After each iteration our search space gets reduced. The proposed approach is not purely FP tree based or rule based pattern growth, it follows the weight tree based pattern growth which is found to execute faster. This also leads to faster execution.

UserSpanPattern is similar to Prefixspan in the way that both work on the reduced database after each scan. Prefixspan is more concentrated in generating frequent patterns than user personalization. The proposed algorithm generates frequent patterns based on user preferences only. This is confirmed by taking weight of frequent patterns, as weights are direct indication of user preference. The study had added the importance of time factor in sequence finding by considering sessions of fixed time slots. Studies at regular intervals helps in finding the mostly preferred pages to be recommended for user and thus enhances personalization which Prefixspan does not ensure. Success rate of recommendations is found to be high in the proposed study. Results and analysis is presented in Table 6.2 of Chapter 6. Prefixspan does not consider ordered sequences. Finding similar groups in the proposed study promotes the possibility of group personalization. Detailed study on clustering and grouping is beyond the scope of this thesis.
5.5.3 Comparison with PLWAP Tree Traversal

PLWAP Tree is an efficient representation method for frequent item sequences using binary codes. The approach follows a pre order traversal of the tree to generate the sequence. Even though the method eliminates the process of intermediate tree generation, it has to scan each frequent subsequence in the pre order way to generate the frequent sequence which is time and memory consuming procedure. The proposed method extracts the mostly frequent category selection based on a particular type of tree traversal which is more efficient. The proposed method finds the frequent occurrences based on the weight and count of similar items from the reduced database each time. Initially the first level is sorted and in most of the cases that makes traversal faster than pre order. After each iteration search space in the proposed method gets reduced. Real dataset comparison with sequence pattern generation methods like PLWAP, FLWAP and FOF Mine is beyond the scope of this thesis and is considered as a future study.

5.6 CONCLUSION

Most of the web sites have highly complicated link structure and this makes extraction usage traversal patterns difficult. Our method predicts future behaviour of user in a site based on similarity generated sequences. Thorough study on sequence generation and clustering is beyond the scope of this work. We concentrate on sequences only for predicting possible future visit. Number of times a user has visited the same category is taken from usage logs in different sessions of equal interval. We follow a session wise and time considered clustering approach for grouping users.

We considered uncertain sequences and partially ordered sequences in our study. Clustering similar usage groups promotes the
possibility of group personalization in our virtual learning environment. We intend to study on how to effectively do recommendations for sites with large clutter. We also wish to extend our study to other fields such as market watch and Bioinformatics.