RESEARCH ARTICLE OPEN ACCESS

An Efficient Info-Gain Algorithm for Finding Frequent Sequential Traversal Patterns from Web Logs Based On Dynamic Weight Constraint

Mr. Ajay Kumar Singh Bais [1], Dr.G.D Gidwani [2]
Department of Computer Science and Engineering
SKSITS
Indore - India

ABSTRACT

Many frequent sequential traversal pattern mining algorithms have been developed which mine the set of frequent subsequences traversal pattern satisfying a minimum support constraint in a session database. However, previous frequent sequential traversal pattern mining algorithms give equal weightage to sequential traversal patterns while the pages in sequential traversal patterns have different importance and have different weightage.

Another main problem in most of the frequent sequential traversal pattern mining algorithms is that they produce a large number of sequential traversal patterns when a minimum support is lowered and they do not provide alternative ways to adjust the number of sequential traversal patterns other than increasing the minimum support. In this paper, we propose a frequent sequential traversal pattern mining algorithm with dynamic weight constraint.

Our main approach is to add the weight constraints into the sequential traversal pattern while maintaining the downward closure property. A weight range is defined to maintain the downward closure property and pages are given different weights and traversal sequences assign a minimum and maximum weight. In scanning a session database, a maximum and minimum weight in the session database is used to prune infrequent sequential traversal subsequence by doing downward closure property can be maintained. Our method produces a few but important sequential traversal patterns in session databases with a low minimum support, by adjusting a weight range of pages and sequence.

The support and confidence are the most popular measures for sequential patterns. The support evaluates frequencies of the patterns and the confidence evaluates frequencies of patterns in the case that sub-patterns are given. These parameters are meaningful and important for some applications. The information gain metric which is widely used in the information theory field, may be useful to evaluate the degree of surprise of the pattern. Target is finding set of patterns that have information gain higher than minimum information gain threshold.

Keywords:- Sequential traversal pattern mining, Weight constraint, Web usage mining, Data mining

I. INTRODUCTION

The World Wide Web is an immense source of data that can come either from the Web content, represented by the billions of pages publicly available, or from the Web usage, represented by the log information daily collected by all the servers around the world. Web Mining is that area of Data Mining which deals with the extraction of interesting knowledge from the World Wide Web [1]. More precisely, Web Content Mining is that part of Web Mining which focuses on the raw information available in Web pages; source data mainly consist of textual data in Web pages (e.g., words, but also tags); typical applications are content-based categorization and content-based ranking of Web pages [2]. Web Structure Mining is that part of Web Mining which focuses on the structure of Web sites; source data mainly consist of the structural information present in Web pages (e.g., links to other pages); typical applications are link-based categorization of Web pages, ranking of Web pages through a combination of content and structure [3], and reverse engineering of Web site models. Web Usage Mining is that part of Web Mining which deals with the extraction of knowledge from server log files; source data mainly consist of the (textual) logs that are collected when users access Web servers and might be represented in standard formats (e.g., Common Log Format, Extended Log Format, LogML)[5]; typical applications are those based on user modeling techniques, such as Web personalization, adaptive Web sites, and user modeling. Figure 1 shows the main application areas of WUM.

Srivastava et al. [6] systematically discuss the development of WUM and classify the content of WUM. Zhang and Liang [7] show the importance of data preprocessing in Web Usage Mining and present an algorithm called "USIA" which boasts high efficiency. Wang and Meinel [8] point out that user behaviors recovery and pattern definition play more important roles in web mining than other applications so they give a new insight on behavior recovery and complicated pattern definition. As current Web Usage Mining applications rely exclusively on the web server log files, Guo et al. [9] propose a system that integrates Web page clustering into log file association mining and use the cluster labels as Web page content indicators in the hope of mining novel and interesting association rules from the combined data source.

Sequential pattern mining has been researched extensively since first introduced by Agrawal [14] in 1995. Before prefix projected sequential pattern mining [15, 16, 17] was developed, Apriori based sequential pattern mining [14, 18] was used based on the downward closure property [14]. That is, if any length k sequential pattern is not frequent in a sequence database, superset sequential patterns can not be frequent. Using this characteristic, Apriori based sequential pattern mining algorithms prune candidate sequential patterns. However, this generates huge candidate sequences and a large amount of the original sequence database must be repeatedly scanned in order to check if a candidate is frequent. This is inefficient and ineffective.

To overcome problems of Apriori based sequential pattern mining algorithms [14, 4, 19], prefix projected sequential pattern growth based approaches [15, 16, 20, 17, 21] have been developed. Sequential pattern growth methods mine the complete set of frequent patterns using a divide and conquer method to reduce the search space without generating all the candidates. There are many limitations exist in the previous sequential traversal pattern mining algorithms. Some are given below.

First, in the real world, some traversal sequences are more important and others are less important. However, previous sequential traversal pattern mining approaches do not consider this characteristic of real datasets. In other words, all pages and traversal sequences are treated uniformly.

Second, most sequential traversal pattern mining algorithms use a support constraint to prune the combinatorial search space. This strategy provides for basic pruning. However, support based pruning is not enough when considering the characteristics of real datasets. In previous mining approaches, after mining datasets to obtain the sequential traversal patterns, there is no way to adjust the number of sequential patterns through user feedback without changing the minimum support. Sequential pattern mining algorithms such as [22, 23, 24] have better performance when a minimum support is high, the database is sparse and the length of the maximum sequential pattern is short. Meanwhile, the main problem with these algorithms is that they can still generate an exponentially large number of sequential patterns and the runtime increases dramatically when a minimum support is lowered. Although closed sequential pattern mining approaches [23, 24] have been used, many sequential patterns are still generated in large dense databases.

We propose an efficient method for sequential traversal pattern mining with weight constraint to tackle these problems of previous traversal pattern mining. Our main goal in this framework is to add weight constraints into the traversal pattern algorithm while keeping the downward closure property. In our approach, a weight range is defined, pages are given different weights to reflect characteristics of the real dataset, and the weights of sequences are calculated. The weight range is used to generate a reasonable number of frequent sequential traversal patterns even in a dense database with a low minimum support. Additionally, both the weight and support of each page are considered separately for pruning the search space. An extensive performance study shows that the number of sequential traversal patterns can be easily adjusted by setting a weight range and the runtime is efficient.

The main contributions of this paper are: 1) introduction of the concept of frequent sequential traversal patterns with weight constraint, 2) classification and incorporation of two key features, a weight and a support, 3) description of frequent traversal pattern mining by using a weight constraint, and 5) Experimental study to compare the performance of our algorithm with a recently developed algorithm, WSpan [26].

The remainder of the paper is organized as follows. In section 2, we describe the problem definition and related work in sequential traversal pattern mining. In Section 3, we develop our algorithm. Section 4 shows experimental results. Finally, conclusions are presented in section 5.

II. PROBLEM DEFINITION AND RELATED WORK

A. Problem definition

Let P = $\{P_l, P_2... P_n\}$ be a unique set of pages. A session S is an ordered list of itemsets, denoted as $(s_l, s_2, .., s_m)$, where s_j is an itemset which is also called an element of the session, and $s_j \subseteq P$. That is, S = $(s_l, s_2, .., s_m)$and s_j is $(x_1\ x_2... x_k)$,where X_t is an item. The brackets are omitted if an itemset has only one item. An item can occur at most one time in an element of a sequence but it can occur multiple times in different elements of a sequence. The size S of a sequence is the number of elements in the sequence. The length, l(s), is the total number of items in the sequence. A sequence with length 1 is called an 1-sequence. A sequence database, D = $\{S_l, S_2, .., S_n\}$, is a set of tuples (sid, s), where sid is a sequence identifier and S_k is an input sequence. A sequence $\alpha= (X_1, X_2, .., X_n)$ is called a subsequence of another sequence $\beta= (Y_1, Y_2, .., Y_m)$ $(n \leq m)$, and β is called a super sequence of α if there exist an integer $1< i_1< ... < i_n < m$ such that $X_1 \subseteq Y_{i1}, ..., X_n \subseteq Y_{in}$. A tuple (sid, S) is said to contain a sequence S_a if S is a super sequence of S_a. The support of a sequence S_a in a sequence database D is the number of tuples in SDB that contains Sa. Given a support threshold, min_sup, a sequence S_a is a frequent sequence in the sequence database if the support of the sequence S_a is no less than a minimum support threshold. The problem of sequential pattern mining is to find the complete set of sequential patterns in the database with a support constraint.

B. Related work

GSP [18] mines sequential patterns based on an Apriori like approach by generating all candidate sequences. This is inefficient and ineffective. To overcome this problem, the database projection growth based approach, FreeSpan [15], was developed. Although FreeSpan outperforms the Apriori based GSP algorithm, FreeSpan may generate any substring combination in a sequence. The projection in FreeSpan must keep all sequences in the original sequence database without length reduction.

PrefixSpan [16], a more efficient pattern growth algorithm was proposed which improves the mining process. The main idea of PrefixSpan is to examine only the prefix subsequences and project only their corresponding suffix subsequences into projected databases. In each projected database, sequential patterns are grown by exploring only local frequent patterns.

In SPADE [25], a vertical id-list data format was presented and the frequent sequence enumeration was performed by a simple join on id lists. SPADE can be considered as an extension of vertical format based frequent pattern mining. SPAM [4] utilizes depth first traversal of the search space combined with a vertical bitmap representation of each sequence. Before SPAM, SPADE and PrefixSpan were two of the fastest algorithms. According to performance evaluations [4], SPAM outperforms SPADE on most datasets and PrefixSpan outperforms SPAM slightly on very small datasets. Except for this case, SPAM outperforms PrefixSpan.

III. PROPOSED WORK

In this section, we suggest an efficient sequential traversal pattern mining algorithm in which the main approach is to apply weight constraints into the frequent ***sequential traversal tree while maintaining the downward closure property***. We discuss our algorithm in detail and show actual examples for sequential traversal pattern mining with weight constraint.

Definition 3.1 Weight Range

A weight of a web page is a non-negative real number that shows the importance of each web page. The weight of each web page is assigned to reflect the importance of each web page in the session database.

Definition 3.2 Traversal sequence with weight

We can use the term, traversal sequence with weight to represent a set of sequential traversal patterns with weight.

Definition 3.2 Average Weight of traversal

We can use the term; average weight of subsequence is the sum of weight all pages in traversal divided by total number of pages in sequence

Definition 3.4 Minimum and Maximum weight of subsequence

Here we define the maximum and minimum weight of traversal is average weight. If the weight of sequence come under the maximum and minimum weight range than given sequence is frequent otherwise infrequent.

A. Sequential traversal pattern with weight constraint

In this paper, pages of traversals are assigned with weights to show their importance. For example, when users traverse web site, they may have different interest in each page, and therefore stay for different times. Web pages can be assigned with a weight standing for the user stay time, frequency of pages, content of pages and type of web site. This paper generalizes the mining problem to the case where pages of traversals are given such weights showing their importance. The weights are taken into account in the measurement of support, the ratio of traversals which contains a candidate pattern. If a page of traversal has a weight which doesn't between the weight ranges then it is removed from session of user and treated as an outlier, and can not consider for the support. For example, when users visit web site, they may traverse through a page very fast to another page, or do another work for a long time during web site visit. This type of page visit is not useful and consider as an outlier because the page is not attentively read by the user.

Table-1. A sequence database as a running example.

Sid	Traversal Sequence	Weight
S1	P2 P1 P3 P4 P1 P5	0.2,0.3,0.12,0.34,0.6,0.3
S2	P1 P2 P4 P3 P4 P2	0.12,0.5,0.91,0.12,0.4,0.26
S3	P1 P2 P1 P3P6 P_7	0.6,0.2,0.32,0.56,0.45,0.7
S4	P2 P3 P6 P5 P1 P4	0.5,0.56,0.32,0.23,0.7,0.54

Table-2. "The example of page with weight range".

S.No.	Page	Support	Weight Range
1	P1	4	0.12 – 0.56
2	P2	4	0.45 – 0.67
3	P3	4	0.23 - 0.67
4	P4	3	0.12 - 0.45
5	P5	2	0.34 – 0.67
6	P6	2	0.24 – 0.8
7	P7	0	0.12 – 0.56

In this section, we propose the concept of sequential traversal patterns with weight constraint, and show their importance.
Example: In session S_1 the weight of P_2 is 0.2 and the support is 4. The weight range for P_2 is from 0.45 to 0.67. So, when we construct the frequent sequential traversal pattern tree P_2 is eliminated from session.

B. Frequent Sequential Traversal Pattern Tree with weight constraint

In this section, a data structure called FSTP-tree is constructed. FSTP-Tree is a data structure, which must satisfy the following conditions. Firstly, it consists one root "null", a set of item prefix subtrees as the children of the root, and a frequent-page head table. Secondly, every page in the page prefix subtree contains three fields: the name of the page, the support of the page, and a link to the next same page. Thirdly, every page in the frequent-page table contains three fields: the name of the page, the weight range and a link to the first node in the tree which denotes this page. The following algorithm to build the FSTP-tree:

Algorithm 1 (FSTP-tree Building: Building a FSTP-tree of the SDB)

Input: A session database SDB, weights of pages and a minimum support

Output: corresponding FSTP-tree

Method:

1. Scan the whole SDB and find frequent pages from SDB based on support and weight range assign to the page. Here we add only those pages that come under the weight range and contribute to the support and those not come in given range consider as outlier and not contribute to support.
2. Create the root of the FSTP-tree, and label it NULL.
3. Scan the whole SDB for the second time. For each session in the SDB, we only preserve the pages which are frequent and have a weight in given weight range, and hold the traversal sequences of pages. The different branches of same prefix can be merged.

C. FSTPMW Algorithm

The divide-and-conquer strategy is used for finding frequent sequential traversal patterns.

To handle the ordered problem, the FSTPMW uses a merging method. Each frequent ordered pattern whose first page is P_1 must be contained in one or more session. The merging process in fact is rebuilding a smaller FSTP-tree. This time, the relative sessions all contains P_1 as the first web page.

The complete algorithm given as:

Algorithm 2 (FSTPMW: Mining frequent sequential traversal pattern)

Input: FSTP-tree

Output: frequent sequential traversal pattern

Method: call FSTPMW (Weight range for each page, support, Minimum & Maximum Weight Range)

```
Procedure FSTPMW (FSTPtreeRootNode node,String prefix )
{
    for each node x in the corresponding page head table do
    if x.support less than minimum support then
    calculate the average weight of prefix
 if minimum weight<=average weight<=maximum weight
{
        output prefix;
}
    return;
    else if i.subs.count == 0 then
    prefix = prefix + i.content;
    calculate the average weight of prefix
    if minimum weight<=average weight<=maximum weight
        {
        output prefix;
        }
    return;
    else
    call CombineTree(i);
    for each node j in i.subs do
    call FSTPMW (j, prefix+i);
    end for
    end if
    end for
   }
```

D. Statistically Significant Patterns

The support and confidence are the most popular measures for sequential patterns. The support evaluates frequencies of the patterns and the confidence evaluates frequencies of patterns in the case that sub-patterns are given. These parameters are meaningful and important for some applications. However, in other applications, the number of occurrences (support) may not always represent the significance of a pattern. Sometimes, a large number of occurrences of an expected frequent pattern may not be as interesting as few occurrences of an expected rare pattern. This pattern called surprising pattern instead of frequent pattern. The information gain metric which is widely used in the information theory field, may be useful to evaluate the degree of surprise of the pattern. Target is finding set of patterns that have information gain higher than minimum information gain threshold. Experiments show that the support threshold has to be set very low to discover a small number of patterns with high information gain.

Note that surprising patterns are anti-monotonic. It means advantage of standard pruning techniques such as Apriori property can't be used. For example, the pattern (I_1,I_2) may have enough information gain while neither $(I_1,*)$ nor $(*,I_2)$ does.

Given a pattern $P=(I_1,I_2,\ldots,I_l)$ and an information gain threshold *min-gain*, the goal is to discover all patterns whose information gain in the sequence S exceed the *min-gain* value. Similar to other parameters in data mining algorithms, the appropriate value of the *min-gain* is application dependent and may be defined by a domain expert. There are some heuristics and methods that user can set the value of this threshold.

Information gain of pattern P is defined as follows:

$$\text{Info-gain}(P)=\text{Info}(P)*\text{Support}(P)$$

$$\text{Info}(P)=\text{Info}(I_1)+\text{Info}(I_2)+\ldots+\text{Info}(I_l)$$

$$\text{Info}(I_k) = -\log_{|I|}{}^{\text{prob}(Ik)}$$

Where prob(I_k) is probability that symbol I_k occurs and |I| is number of items or events in S.

4. Analysis and Performance evaluation

In this section, we present our performance study over various datasets. We report our experimental results on the performance of FSTPMW in comparison with a recently developed algorithm; WSpan [26], which is the fastest algorithm for mining sequential patterns. The main purpose of this experiment is to demonstrate how effectively the sequential traversal patterns with weight constraint can be generated by incorporating a weight page, weight of sequence with a support. First, we show how the number of sequential traversal patterns can be adjusted through user assign weights, the efficiency in terms of runtime of the FSTPMW algorithm, and the quality of sequential traversal patterns. Second, we show that FSTPMW has good scalability against the number of sequence transactions in the datasets.

4.1 Environmental results. Comparison of FSTPMW and WSpan

In this performance test, we focused on the efficiency of using a weight range. Our experiment shows that in most cases, FSTPMW outperforms WSpan. First, we evaluate the performance on the kosarak dataset.

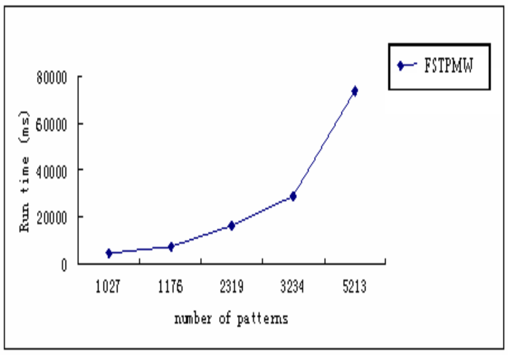

Figure-1. Scalability with number of frequent sequential traversal patterns

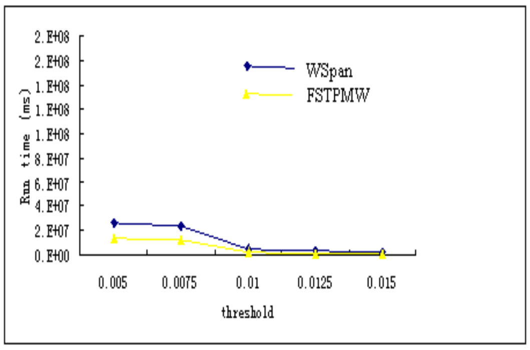

Figure-2. Runtime

Support/ Item	ITEM-1	ITEM-2	ITEM-3	ITEM-4
3	0.50	0.17	0.17	0.17
6	0.56	0.22	0.11	0.11
10	0.50	0.25	0.08	0.17
20	0.50	0.33	0.00	0.17
30	0.58	0.17	0.08	0.17

Table 3 : Probability Gain with Different Support on Items

Support/ Item	ITEM-1	ITEM-2	ITEM-3	ITEM-4
3	1.50	1.29	1.29	1.29
6	2.12	2.17	1.58	1.58
10	3.00	3.00	1.79	2.58
20	1.50	1.58	0.00	1.29
30	2.72	2.58	1.79	2.58

Table 4 : Info-Gain with Different Support on Items

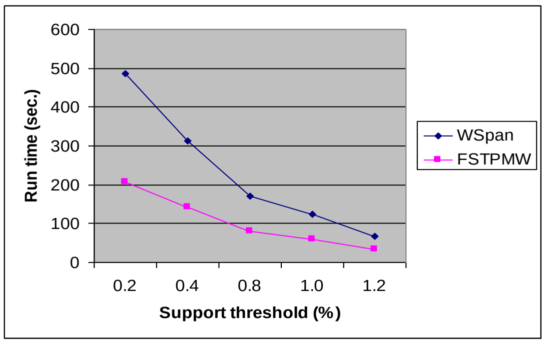

Figure-3 : Scalability of FSTPMW and WSpan with threshold on BMS-WebVies-1

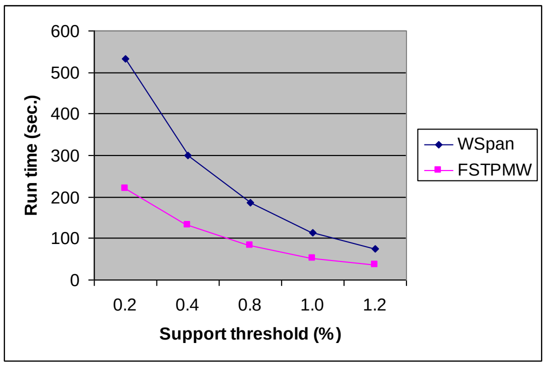

Figure -4 : Scalability of FSTPMW and WSpan with threshold on BMS-WebVies-2

4.2 Further extension

FSTPMW basically focuses on sequential pattern mining with weight constraint uses a weight range to adjust the number of sequential traversal patterns. Frequent sequential traversal pattern mining can be extended by considering levels of support and/or weight of sequential traversal patterns. There are many areas in which items have different importance and patterns with a similar level of support and/or weight are more meaningful. For example, the concept of strong support and/or weight affinity can be applied in DNA analysis. We can give importance to specific DNA patterns and find interesting DNA patterns. Our algorithm can be extended by considering the levels of support and/or weight of sequential traversal patterns. By not only giving a balance between the two measures of support and/or weight, but also considering both support and/or weight affinity between items within patterns, more valuable sequential traversal patterns can be generated.

V. CONCLUSION

Many studies exist on mining sequential frequent patterns. One of the main limitations of the traditional approach for mining sequential traversal patterns is that all items are treated uniformly, while each page of web site has different importance. Moreover, previous sequential traversal pattern mining generates a very large number of subsequence as the minimum support becomes lower. In this paper, we developed FSTPMW which focused on frequent sequential traversal pattern mining with weight constraint. A weight range is used to adjust the number of sequential patterns. The extensive performance analysis shows that FSTPMW is efficient and scalable in mining sequential traversal patterns.

REFERENCES

[1] Etzioni, O. (1996). The world-wide Web: quagmire or gold mine? Communications of the ACM, 39 (11), 65–68.

[2] Kosala, R and Blockeel, H. (2000). Web mining research: a survey. SIGKDD Explorations, July, 2 (1), 1-15.

[3] Brin, S and Page,L. (1998). The anatomy of a large-scale hyper-textual Web search engine. Computer Networks and ISDN Systems, 30 (1–7), 107–117.

[4] J. Ayres, J. Gehrke, T. Yiu, and J. Flannick, "Sequential Pattern Mining using A Bitmap Representation, SIGKDD'02, 2002.

[5] J.R. Punin, M.S. Krishnamoorthy, and M.J. Zaki. (2001). LOGML - Log Markup Language for Web Usage Mining, in WEBKDD Workshop 2001: Mining Log Data Across All Customer TouchPoints (with SIGKDD01), San Francisco, August, pp. 88–112.

[6] Srivastava J , Cooley R , and Mukund Deshpanda. (2000). Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data. SIGKDD Explorations, 1 (2), 12-23.

[7] Zhang Huiying and Liang, Wei. (2004). An intelligent algorithm of data pre-processing in Web usage mining, Proceedings of the World Congress on Intelligent Control and Automation (WCICA), v 4, WCICA, p3119-3123.

[8] Long Wang. (2004). Christoph Meinel. Behaviour Recovery and Complicated Pattern Definition in Web Usage Mining. Web Engineering: 4th International Conference, ICWE 2004, Munich, Germany, July 26-30, pp. 531 – 543.

[9] Guo, Jiayun, Keelj, Vlado, and Gao, Qigang. (2005). Integrating Web Content Clustering into Web Log Association Rule Mining, Advances in Artificial Intelligence: 18th Conference of the Canadian Society for Computational Studies of Intelligence, Canada, May 9-11, pp. 182

[10] Agrawal, R and Srikant, R. (1994). Fast algorithms for mining association rules, Proc. of the 20th international Conference on very large database, Chile, 487-499.

[11] Park J S, Chen M -S, and Yu P S. (1995). An effective Hash-based algorithm for mining association rules, Proceedings of 1995 ACM-SIGMOD International Conference on Management of Data (SIGMOD'95). San Jo se, CA, 175-186.

[12] Savasere A ,Omiecinski E, and Navathe S. (1995). An efficient algorithm for mining association rules in large databases . VLDB'95 , 432-443.

[13] Toivonen H. (1996). Sampling Large Databases for Association Rules, Proceedings of 22th VLDB Conf. Bombay, India, 134-145.

[14] R. Agrawal, and R. Srikant, "Mining Sequential Patterns, " ICDE, 1995.

[15] J. Han, J. Pei, B. Mortazavi-Asi, Q. Chen, U. Dayal, M. C. Hsu, "FreeSpan: Frequent Pattern-Projected Sequential Pattern Mining, " SIGKDD'00, 2000

[16] J. Pei, J. Han, B. Mortazavi-Asi, H. Pino, "PrefixSpan: Mining Sequential Patterns Efficiently by Prefix- Projected Pattern Growth," ICDE'01, 2001.

[17] H. Pinto, J. Han, J. Pei, K. Wang, "Multi-dimensional Sequence Pattern Mining, " CIKM'01, 2001.

[18] R. Srikant, and R. Agrawal, "Mining Sequential Patterns: Generalizations and Performance Improvements, EDBT, " 1996.

[19] M. Garofalakis, R. Rastogi, and K. Shim, "SPIRIT: Sequential pattern mining with regular expression constraints, " VLDB'99, 1999.

[20] J. Pei, J. Han, J. Wang, H. Pinto, Q. Chen, U. Dayal, M. C. Hsu, "Mining Sequential Patterns by Pattern-Growth: The PrefixSpan Approach, " IEEE Transactions on Knowledge and Data Engineering, Oct, 2004.

[21] M. Seno and G. Karypis, "SLPMiner: An Algorithm for Finding Frequent Sequential Patterns Using

Length-Decreasing Support Constraints," ICDM'02, 2002.

[22] J. Pei, J. Han, and W. Wang, "Mining Sequential Patterns with Constraints in Large Databases," ACM CIKf, Nov. 2002.

[23] J. Wang, and J. Han, "BIDE: Efficient Mining of Frequent Closed Sequences, ICDE'04, 2004.

[24] X. Yan, J. Han, R. Afshar, "CloSpan: Mining Closed Sequential Patterns in Large Datasets, " SDM'03, 2003.

[25] M. Zaki, "SPADE: An efficient algorithm for mining frequent sequences. Machine Learning, " 2001.

[26] U. Yun and J.J. Leggett, "WSpan: Weighted Sequential Pattern Mining in Large Sequence Databases," *Proc. Of the Third Int'l Conf. on IEEE Intelligent Systems*, Sep. 2006, pp. 512-517.

[27] U. Yun and J.J. Leggett, "WFIM: Weighted Frequent Itemset Mining with a Weight Range and a Minimum Weight," *Proc. Of the Fifth SIAM Int'l Conf. on Data Mining*, Apr. 2005, pp. 636- 640.

[28] U. Yun and J.J. Leggett, "WLPMiner: Weighted Frequent Pattern Mining with Length-Decreasing Support Constraints," *Proc. Of* the *9th Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD`05)*, May 2005, pp. 555-567.

[29] U. Yun, "Mining Lossless Closed Frequent Patterns with Weight Constraints," *Knowledge Based Systems*, vol. 20, Feb. 2007, pp. 86-97.

[30]. Mahdi Esmaeili and Fazekas Gabor, "Finding Sequential Patterns from Large Sequence data," IJCSI, vol. 7, Issue 1, Jan 2010, pp. 43-46.