

A Resourceful Info-Gain System Designed for Acquiring Frequent Sequential Traversal Outlines from Web Logs based on Dynamic Weight Constraint

Rahul Moriwal¹ and Dr. Anil Rao Pimpalapure²

¹⁻²Eklavya University Damoh (M.P.) INDIA

Email: rmoriwal@gmail.com, pimpalapureanil@gmail.com

Abstract— Numerous methods for mining frequent sequential traversal trends have been created. These algorithms mine the set of often occurring subsequences traversal trends in a session database that fulfill a minimal support criterion. Nevertheless, prior frequent sequential traversal pattern mining methods assign a uniform weight to sequential traversal trends, despite the fact that the pages inside these trends varied in significance and weight. Another major issue with the majority of frequently occurring sequential traversal pattern mining techniques is that, when a minimum support is lowered, they generate a lot of sequential traversal patterns and do not offer any other options for adjusting the number of sequential traversal patterns besides raising the minimum support. Our proposal for a frequent sequential traversal pattern mining method with dynamic weight constraint is presented in this work.

Our major strategy is to preserve the downward closure feature while incorporating the weight limitations into the sequential traversal trend. To preserve the downward closure property, a weight range is established, pages are assigned varying weights, and traversal sequences designate a minimum and maximum weight. A highest and lowest weight in the session database are used to trim rare sequential traversal subsequences during database scanning so that the downward closure attribute may be preserved. By varying the weight range of pages and sequence, our approach generates a few but significant sequential traversal trends in session databases with a low minimum support. For sequential trends, the most often used metrics are confidence and backing. When sub-trends are provided, the confidence assesses trends frequencies while the support evaluates trends frequencies. For some applications, these factors have significance and importance. To assess the level of surprise of the trends, the information gain metric—which is frequently employed in the field of information theory—might be helpful. The goal is to identify a group of trends whose information gain exceeds the minimal criterion.

Keywords — Sequential traversal pattern mining, Weight constraint, Web usage mining, Data Mining

I. INTRODUCTION

The World Wide Web is a vast data stream that can originate from two sources: the billions of pages that are publicly accessible as evidence of web content, or the daily log information that all servers worldwide gather as evidence of web usage.

Web mining is the branch of data mining that focuses on locating and analyzing useful information on the World Wide Web [1]. More specifically, Web Content Mining is the area of Web Mining that concentrates on the unprocessed data found in Web pages; source data primarily consists of textual data found in Web pages (such as words and tags); common uses are content-based ranking and categorization of Web pages [2]. Web structure mining is the branch of web mining that focuses on the structure of websites; the source data mostly consists of the structural information found on websites (such as links to other pages); common uses include ranking websites based on a mixture of content and structure [3], categorizing websites based on links, and reverse engineering website models. Web Usage Mining is the branch of Web Mining that deals with knowledge extraction from server log files; source data mostly consists of the (textual) logs that are collected, sometimes in standard formats (e.g., Common Log Format, Extended Log Format, LogML), when users access Web servers.[5]; common uses include Web customization, adaptable websites, and user modeling, which depends on user modeling strategies. The primary WUM application areas are depicted in Figure 1.

The evolution of WUM is methodically discussed and its content is categorized by Srivastava et al. [6]. In addition to highlighting the value of data preparation for Web Usage Mining, Zhang and Liang [7] propose the very efficient "USIA" method. Wang and Meinel [8] provide fresh insight into behavior recovery and complex pattern definition by pointing out that user behaviors and pattern definition are more significant in web mining than in other applications.

In order to mine new and intriguing association rules from the combined data source, Guo et al. [9] propose a system that combines Web page clustering into log file association mining and uses the cluster labels as Web page content indicators. This is because current Web Usage Mining applications only rely on the web server log files.

Since its initial introduction by Agrawal [14] in 1995, sequential pattern mining has been the subject of much study. Based on the downward closure property [14], Apriori based sequential pattern mining [14, 18] was employed prior to the development of initial projected sequential pattern mining [15, 16, 17]. In other words, superset sequential patterns cannot be common in a sequence database if any length k sequential pattern is not. Apriori-based sequential pattern mining techniques use this property to reduce the number of potential sequential patterns. Nevertheless, this produces enormous candidate sequences, and to determine if a candidate is common, a sizable portion of the original sequence database needs to be regularly examined. This is both ineffectual and inefficient.

Prefix projected sequential pattern growth-based techniques [15, 16, 20, 17, 21] have been developed to address issues with Apriori based sequential pattern mining algorithms [14, 4, 19]. Divide and conquer techniques are used in sequential pattern growth methods to mine the whole collection of frequently occurring patterns, reducing the search space without producing all of the possibilities. The earlier sequential traversal pattern mining techniques have several drawbacks. A few are listed below.

Firstly, certain traversal sequences are more significant in the actual world than others. Nevertheless, this feature of actual datasets is not taken into account by earlier sequential traversal pattern mining techniques. Stated differently, every page and traversal sequence receive the same treatment.

Secondly, a support constraint is used by the majority of sequential traversal pattern mining algorithms to reduce the size of the combinatorial search space. This plan allows for simple pruning. Support-based pruning, however, falls short when genuine dataset features are taken into account. Prior mining techniques did not allow for user feedback to modify the minimal support in order to modify the number of sequential patterns that were obtained after mining datasets to produce sequential traversal patterns. When the database is sparse, the maximum sequential pattern's length is short, and the minimum support is large, sequential pattern mining techniques like those found in [22, 23, 24] perform well. However, the primary issue with these algorithms is that even at low minimum supports, they may still produce an exponentially high number of sequential patterns, and this leads to a sharp rise in runtime. Despite the adoption of closed sequential pattern mining algorithms [23, 24], big dense databases continue to create a high number of sequential patterns.

To address these issues with earlier traversal pattern mining, we suggest an effective technique for sequential traversal pattern mining with weight constraints. Our primary objective with this framework is to maintain the downward closure condition while including weight limitations into the traversal pattern algorithm. Our method determines a weight range, assigns distinct weights to pages based on real dataset properties, and computes sequence weights. Even in a large database with a low minimum support, the weight range is employed to construct a decent number of frequent sequential traversal patterns. Furthermore, while narrowing down the search space, each page's weight and support are taken into independent consideration.

A detailed performance investigation demonstrates that the runtime is efficient and that it is simple to modify the amount of sequential traversal patterns by specifying a weight range.

The notion of prevalent sequential traversal patterns with weight constraint is introduced in this paper, which also includes two important features—weight and support—in its classification and incorporation. It also describes frequent traversal pattern mining through the use of a weight constraint and conducts an experimental study that examines the performance of our algorithm with that of a recently developed algorithm, WSpan [26]. This is how the rest of the work is structured. We define the topic and discuss related work in sequential traversal pattern mining in section 2. We design our algorithm in Section 3. The findings of the experiment are shown in Section 4. Lastly, section 5 presents conclusions.

II. PROBLEM DEFINITION & RELATED WORK

A. Problem Definition

Consider a unique collection of pages $P = \{P_1, P_2, \dots, P_n\}$. A meeting s_j is an a set of items, also known as an element of the session, and $s_j \subseteq P$. S is an ordered list of sets of items, represented as $(s_1, s_2, \dots, s_m)$. $S = (s_1, s_2, \dots, s_m)$ and $s_j = (x_1 x_2 \dots x_k)$, where X_t is an object, are what this means. If there is only one item in an itemset, the braces are removed. An item may appear in a sequence element only once, but it may appear more than once in several sequence elements. A sequence's size S is equal to the total number of its elements. The total number of elements in the sequence is the length, or $l(s)$. A 1-sequence is a sequence that has a length of one. A set of tuples (sid, s) , where sid is a sequence identifier and S_k is an input sequence, makes up a sequence database, $D = \{S_1, S_2, \dots, S_n\}$. When an integer $1 < i_1 < \dots < i_n \leq m$ exists such that $X_1 \subseteq Y_{i_1}, \dots, X_n \subseteq Y_{i_n}$, then $\alpha = (X_1, X_2, \dots, X_n)$ is a subsequence of another sequence $\beta = (Y_1, Y_2, \dots, Y_m)$ ($n \leq m$). In this case, β is referred to as a super sequence of α .

If S is a super sequence of S_a , then a tuple (sid, S) is said to contain a sequence S_a . The number of tuples in SDB that include a sequence S_a indicates the support of that sequence in sequence database D . If the support of a sequence S_a is more than or equal to a minimal support threshold, min_sup , then the sequence S_a is considered common in the sequence database. Finding every sequential pattern in a database while adhering to a support constraint is the challenge of sequential pattern mining.

B. Related Work

GSP [18] generates all potential sequences in order to mine sequential patterns using a method akin to Apriori. This is both ineffectual and inefficient. The database projection growth based method known as FreeSpan [15] was created in order to solve this issue. FreeSpan may produce every substring combination in a sequence, even though it performs better than the Apriori-based GSP technique. All sequences in the original sequence database must be preserved in the FreeSpan projection without any length reduction.

A more effective pattern growth technique called PrefixSpan [16] was put up to enhance the mining procedure. PrefixSpan's primary goal is to study just the prefix subsequences and project onto projected databases only the appropriate suffix subsequences. Sequential patterns are developed in each projected database by examining just local frequent patterns. A vertical id-list data structure was used in SPADE [25], and a straightforward join on id lists was used to enumerate common sequences. SPADE may be viewed as a progression of frequent pattern mining based on vertical formats. In SPAM [4], each sequence is represented as a vertical bitmap, and the search space is traversed depth first. Two of the quickest algorithms prior to SPAM were SPADE and PrefixSpan. Performance assessments [4] show that on most datasets, SPAM performs better than SPADE, while on extremely small datasets, PrefixSpan performs marginally better than SPAM. PrefixSpan is not as good as SPAM, with one exception.

III. PROPOSED WORK

In this part, we propose an effective sequential traversal pattern mining technique, where the downward closure condition is maintained while applying weight restrictions to the regular sequential traversal tree. We go into great depth about our technique and provide real-world examples of sequential traversal pattern mining underweight constraints.

Definition A. Weight Range

A web page's weight is a real number that is non-negative and indicates how important each page is. Each webpage is given a weight that corresponds to its significance inside the session database.

Definition B. Traversal sequence with weight

A collection of sequential traversal patterns with weight can be represented by the phrase traversal sequence with weight.

Definition C. Average Weight of traversal

The average weight of a subsequence may be expressed as follows: the total weight of all the pages in the traversal divided by the total amount of pages in the sequence.

Definition 3.4 Minimum and Maximum weight of subsequence

Here, average weight is used to establish the highest and lowest traversal weights. The sequence is frequent if its weight falls between the highest and lowest weight ranges; otherwise, it is infrequent

A. Sequential Traversal Pattern with Weight Constraint

In this paper, pages containing traversals are given weights in this document to indicate their significance. For instance, visitors may visit a website and remain for varying lengths of time depending on their level of interest on each page. Websites can be ranked according to factors such as user duration, page frequency, content, and website type. The mining problem is generalized in this study to the situation where pages of traversals are assigned weights that indicate their significance.

The ratio of traversals containing a candidate pattern is used to quantify support, and this measurement takes the weights into consideration. A traversal page that falls outside of the weight ranges is deleted from the user's session, flagged as an anomaly, and is not eligible for assistance. For instance, when visitors visit a website, they could quickly navigate from one page to another or spend a significant amount of time working on something else. Because the user is not carefully reading the website, this kind of page visit is considered abnormal and is not helpful.

TABLE 1. A SEQUENCE DATABASE AS A RUNNING EXAMPLE

Sid	TraversalSequence	Weight
S1	P2 P1 P3P4 P1 P5	0.2,0.3,0.12,0.34,0.6,0.3
S2	P1 P2 P4P3 P4 P2	0.12,0.5,0.91,0.12,0.4,0.26
S3	P1 P2 P1P3P6 P ₇	0.6,0.2,0.32,0.56,0.45,0.7
S4	P2 P3 P6P5 P1 P4	0.5,0.56,0.32,0.23,0.7,0.54

TABLE II. 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

We introduce and illustrate the notion of sequential traversal patterns with weight constraint in this section.

For instance: P2's weight in session S1 is 0.2, and the support is 4. P2 has a weight range of 0.45 to 0.67. Thus, when we

P2 is removed from the session after constructing the frequent sequential traversal pattern tree.

B. Frequent Sequential Traversal Pattern Tree with weight constraint

The construction of the FSTP-tree data structure is covered in this section. The FSTP-Tree data structure has to meet the following requirements. It is composed of one root called "null," a number of item prefix subtrees that are the root's descendants, and a frequent-page head table. Second, each page in the page prefix subtree has three fields: the page name, the page support, and a link to the page that comes after it. Thirdly, each page in the frequently visited page table has three fields: the page name, the weight range, and a link to the page's initial node in the tree. The algorithm used to create the FSTP-tree is as follows:

Algorithm 1 (FSTP-tree Construction: Constructing an SDB FSTP-tree)

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

Output: corresponding FSTP-tree

Method

- Look through the whole SDB and identify frequently used pages by using the support and weight range assigned to each page. Only pages that fall within the weight range and provide support are added here; sites that do not fall within the range are regarded as outliers and do not provide support.
- Create the root of the FSTP-tree, and label it NULL.
- Perform a second scan of the whole SDB. We put on the traversal sequences of pages and only retain the pages that are often accessed and fall within a certain weight range for each session in the SDB. One can combine distinct branches belonging to the same prefix.

C. FSTPMW Algorithm

The technique known as "divide and conquer" is employed to identify recurring sequential traversal patterns. The merging approach is used by the FSTPMW to solve the ordered issue. Every common ordered pattern with a P1 initial page has to be included in at least one session. In actuality, the merging procedure rebuilds a smaller FSTP-tree. This time, P1 is the initial webpage in each of the related sessions.

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, Lowest & Highest 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

For sequential patterns, the most often used metrics are support and confidence. When sub-patterns are provided, the confidence assesses pattern frequencies while the support evaluates pattern frequencies. For some applications, these factors have significance and importance. The quantity of occurrences (support), however, could not always indicate the importance of a pattern in other contexts. A few occurrences of an expected uncommon pattern may occasionally be more fascinating than many occurrences of an expected frequent pattern. This pattern is not a regular pattern; rather, it is a surprise pattern. This pattern is not a regular pattern; rather, it is a surprise pattern. To assess the level of surprise of the pattern, the information gain metric—which is frequently employed in the field of information theory—might be helpful. The goal is to identify a group of patterns whose information gain exceeds the minimal criterion. It is demonstrated by experiments that in order to find a limited number of patterns with substantial information gain, the support threshold must be set very low.

Keep in mind that unexpected patterns are not monotone. It implies that benefits of common trimming methods, such the Apriori characteristic, cannot be utilized. For example, the pattern (I ,I) may have enough information gain while neither (I ,*) nor (*,I) does. Given a pattern $P=(I ,I ,...,I)$ 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: $\text{Info-gain}(P) = \text{Info}(P) * \text{Support}(P)$

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

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

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

IV. ANALYSIS AND PERFORMANCE EVALUATION

In this part, we show our performance analysis across several datasets. We provide our preliminary findings on the outcome of FSTPMW in relation to WSpan [26], a recently developed method that is the quickest for mining sequential trends. The primary goal of this project is to show how successfully sequential traversal trends with weight constraints can be constructed by combining a weight page, weight of sequence, and support. First, we demonstrate how the amount of sequential traversal patterns may be modified using user-assigned weights, the FSTPMW algorithm's runtime efficiency, and the quality of sequential traversal trends. Second, we demonstrate that FSTPMW has high scalability in terms of sequence transactions in the datasets.

A. Environmental Results: Comparison of FSTPMW and WSpan

In this performance assessment, we looked at the effectiveness of utilizing a weight range. Our experiment demonstrates that FSTPMW consistently beats WSpan. First, we analyze accuracy on the Kosarak dataset.

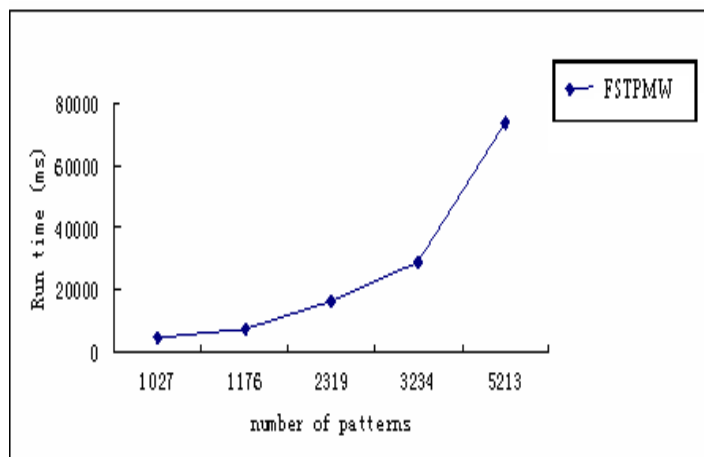


Figure-1. Scalability with number of frequent sequential traversal patterns

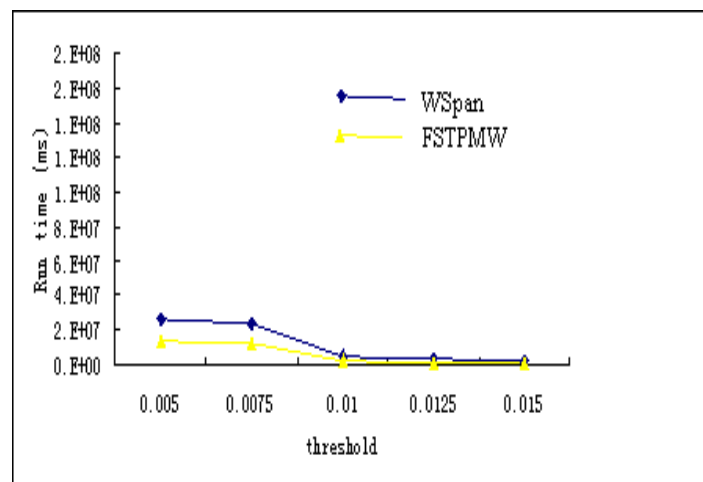


Figure-2. Runtime

TABLE III. PROBABILITY GAIN WITH DIFFERENT SUPPORT ON ITEMS

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 IV. INFO-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

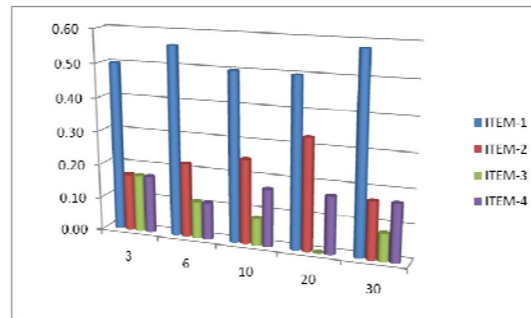


Figure-3 Patterns Utility Graph for finding Probability of Items of Page

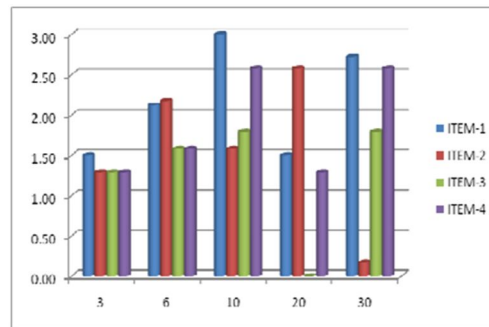


Figure -4. Patterns Utility Graph for finding Information Gain of Items of Page

B. Further extension

FSTPMW primarily focuses on sequential trends mining with weight constraints, which use a weight range to control the amount of sequential traversal trends. Frequent sequential traversal trends mining can be expanded to include degrees of support and/or weight of sequential traversal trends. In many cases, objects have varying value, and trends with comparable levels of assistance and/or weight are more significant. For example, the idea of solid backing and/or weight friendliness can be used in DNA analysis. We may prioritize certain DNA patterns and discover intriguing DNA trends. Our technique may be expanded by taking into account the degrees of assistance and/or weight of sequential traversal trends. More beneficial sequential traversal trends can be constructed by not only balancing the two measures of assistance and/or weight, but also taking into account the assistance and/or weight friendliness between items inside patterns.

V. CONCLUSION

Many research have been conducted on extracting sequential frequent patterns. One of the primary drawbacks of the old technique of mining sequential traversal trends is that all objects are handled similarly, but each page of a website has a varied relevance. Furthermore, earlier sequential traversal trends mining produces a significant

number of subsequences when the minimal support decreases. In this study, we provide FSTPMW, which focuses on regular sequential traversal trends mining underweight constraints. A weight range is used to control the number of successive patterns. The detailed performance research demonstrates that FSTPMW is fruitful and flexible in mining sequential traversal trends.

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] JM. 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 Esmaili and Fazekas Gabor, "Finding Sequential Patterns from Large Sequence data," *IJCSI*, vol. 7, Issue 1, Jan 2010, pp. 43-46.