An Enhancement for finding User Behavior on Progressive Data Using Weight Constraint

[1]Dhirendra Kumar Jha, [2]Ketan Singh
[1,2]Malhotra Technical Research Institute, Bhopal, India
Email: dhirendrajha@gmail.com, ketansingh26@gmail.com

Abstract: ***Every Organizations need to understand their customer's behavior, preferences and future needs every time depend on past behavior. Web Usage Mining is an active research topic in which user session clustering is done to understand user's activities. I am proposing an enhancement to the web log mining process based on online navigational pattern prediction. In this paper, I use Neural based approach Self Organizing Mapping for clustering of session as a trend analysis with some parameters. It depends on the performance of the clustering of the number of requests. Here I am using SOM algorithm in Frequent Sequential Traversal Pattern Mining called STPMW. By proceeding this way, first I use SOM algorithm and getting some cluster of web-logs. Here we load that web-log cluster which is nearly related to frequent pattern. After that I am applying Min-Max Weight of Page in Sequential Traversal Pattern. If given support comes between min and max of weight range so item is frequent else I check average weight. Finally I am establish good prediction with quantity of data and the quality of the results.***

Keywords – Web Usage Mining; Sequential Patterns; sequence Tree; Web Log Data; Web Services; Neural Network; Clustering.

I. INTRODUCTION

Organizations, companies and institutions are relying more and more on their websites to interact with customers. Retaining current customers and attracting potential ones push these organizations, companies and institutions to come across for striking ways to make their websites more useful and efficient. The WWW [2] is an immense source of data that can come either from the Web content, represented by the billions of pages openly available, or from the Web usage, represented by the register information daily collected by all the servers around the world. Web Mining is that part of Data Mining which deals with the extraction of interesting knowledge from the World Wide Web. Web usage mining [3] has many applications, e.g., personalization of web substance, support to the design, recommendation systems, pre-fetching and caching etc. There are various benefits of web usage mining, especially in e-commerce. Customers can be targeted with apposite advertisement. Also, related products can be recommended to customers in real-time while browsing the website. According to, the usage mining process can be divided into three steps. It starts first with data cleaning and pre-processing. Second, the pre-processed data is mined for some hidden and constructive information. At last, the web log mining process ends by analyzing the mining results.

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 logs that are collected when users access Web servers and might be represented in standard formats (e.g., Common & Extended Log Format, LogML)[6]; classic application are those based on user modeling method, such as Web personalization, acclimatize Web sites, and user modeling. As current Web Usage Mining applications rely exclusively on the web server log files, Guo et al. 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.

Kohonen Self-Organizing Maps [10] developed in 1982 by Tuevo Kohonen, a professor emeritus of the Academy of Finland [Wiki-01]. Self-Organizing Maps are aptly named [8]. "Self Organizing" is because no direction is requir ed. SOMs learn on their own through unsupervised competitive learning. "Maps" is because they attempt to map their weights to conform to the given input data. The nodes in a SOM network attempt to become like the inputs presented to them. In this common sense, this is how they study. It is also be called "Feature Maps", as in Self-Organizing Feature Maps [10]. Retaining principle 'features' of the input data isa fundamental principle of SOMs, and it is one of the things that makes them so precious. Specially, the topological relationships between input data are preserved when mapped to a SOM network. This has a practical value of representing complex data. Self Organizing Map (SOM) are those which have processes that allow automatically the internal organization to grow without being guided or controlled by any external source. SOM usually show emergent properties which allow, starting from simple rules, to obtain complex structures. Self organizing concept is a basis in the description of biological systems, from sub-cellular level to ecosystem level. On the other hand, areas such as cybernetics, cellular automata, random graphs, evolutionary computing and artificial life, present self organizing features.

II. PROBLEM DEFINITION

Frequent sequence mining in big data is an important and yet a challenging data mining work. Frequent sequence mining has been analyzed for more than a 20 decade and has become an important component of many prediction or recommendation systems, e.g. The online user predicting which web pages, likely to visit or predicting which products the online user is likely to buy together. All algorithms used for frequent sequence mining could be classified either as exact or approximate algorithms. Accurate frequent sequence mining algorithms usually read the whole database several times, and if the database is very large, then frequent sequence mining is not compatible with limited availability of computer resources and real time constraints. Also in some cases exact methods cannot be used for continuous data streams, because the amount of data is typically so huge that it becomes difficult to store it and almost impossible to fully read it and

process on time when the decision maker needs it. However in some cases where a precise result is required, then exact methods are irreplaceable by any approximate methods, because the precision is very essential. For example, in biological databases an important task is similarity search and comparison between diseased and healthy tissues to identify critical differences between the two classes of genes. This can be done only using exact frequent pattern methods in order to retrieve the gene sequences from the two tissue classes and compare the frequently occurring patterns of every class. Generally sequences occurring more frequently in the diseased samples than in the healthy samples might indicate the genetic factors of the disease (Han, 2002).

Approximate methods are very popular in analyzing data where the computation speed of the algorithm is more important than the precision and some classification errors into frequent and rare sequences are acceptable. Trading in the Foreign Exchange is a good example in order to show the importance of quick approximate algorithms for frequent sequence mining.

The knowledge of frequently occurring sequences is one of techniques being used in trading strategy and with the combination of other trading techniques the decisions in trading strategy could be taken when to open a trading position. There is a huge amount of data captured every second in the financial markets, e.g. currency exchange data, stock market, etc. Traders often use technical analysis for forecasting the direction of prices through the study of earlier period market data, mainly price and volume. There are various techniques in technological analysis, e.g. candlestick charting, Dow Theory, etc. and many traders combine elements from more than one technique. Some traders use a very subjective judgment to decide which pattern a price rate reflects at a given time and what the interpretation of that pattern should be. Others use a strictly mechanical approach to pattern identification and interpretation and search for archetypal price graph patterns, such as the well known head and shoulders, double top/bottom reversal patterns or different market indicators, which are mathematical transformations of price. These patterns and indicators are used to help assessing whether an asset is trending, and if it is, then predict of its direction. Immediate results from historical stock market data are needed for traders to estimate whether it is worth to enter or exit trading positions, therefore fast approximate algorithms are the key as none of exact algorithms can be used due to the continuous data stream. Any improvement in time taken for probabilistic algorithms is a big win, because every millisecond is very important to gain more profit in the trading strategy.

Various approximate sequence mining algorithms were proposed recently which are much faster than accurate algorithms, though do not have the theoretical mistake estimation of algorithm results and this leads to a number of empirical tests being performed in order to get the empirical evidence of errors made identifying frequent sequences. Generally such algorithms are sensitive to the data that is used for finding frequent sequences and the experiment results might vary significantly depending on the data. The theoretical estimation of errors made identifying frequent sequences gives a huge advantage for data analysis because then it is known what precision might be used when finding frequent sequences in large databases.

Therefore, the goal of this study is to propose novel approximate frequent sequence mining algorithms with theoretically defined error probabilities and to compare their performance with other exact and approximate approaches. In addition to frequent sequence mining, a visual representation technique for large information is proposed. It is extremely important for decision making when studying behaviour in the Internet users.

III. TASKS AND OBJECTIVES OF THE RESEARCH

In this study the objectives are:

(1) Create new proposed frequent sequence mining algorithms with SOM for which theoretical estimation of induced errors can be made;

(2) Propose an approach to create good clustering and prediction with quality of the results for caching to improve response time.

In pursuit of these objectives, the specific work is:

- Study the existing frequent sequence pattern mining algorithms [13](both exact and proposed);
- Propose novel frequent sequence mining method [9] using SOM with an estimate of the probability of error made by this method when classifying the sequences as frequent or rare;
- Construct a sample of an original database for finding frequent sequences by proposed method;
- Propose random sampling based methods for mining frequent sequences after getting clusters by using SOM[10], with the estimated probabilities of errors made by these methods;
- After that we select some cluster, based on condition and group it.
- Now, propose a method on this group of clusters that relies upon the Markov property for Proposed mining frequent sequences;
- calculate the performance of proposed procedure and compare the results with other exact and Proposed frequent sequence mining approaches;
- Visually represent the analysis of performance of Internet user using various visualization methods and establish good prediction with quantity of data and the quality of the results.

IV. RELATED WORK

In WUM [3] in general it is not required to know about a user's identity; however, it is necessary to distinguish between different users. If a website requires users to sign in before they can start browsing, it will be very easy not only to differentiate between users but also to identify each single user. The problem arises when a website allows visitors to anonymously browse its content, which is common place. In this case, relying only on what the web server records in the

log to differentiate between visitors becomes challenging. In fact, it becomes more challenging if the web server is logging visitors activities using common log format. The difficulty in distinguishing between visitors arises from the fact that some visitor's activities will be logged with the same IP address due to transiting by the same proxy server or by sharing an internet connection. Different ways to distinguish between users has been listed in [6]. By using cookies users can be identified. When a visitor connects for the first time to a website that uses cookies, the server sends a cookie to the web client along with the requested resource. The next time the same user requests another resource from the same website, the browser will send the cookie stored on the visitors computer along with the request if the cookie is still not expired. The server will be able to recognize the user if the request is coming with a cookie. The problem with this approach is that users can delete cookies stored on their computers and the server will not be able to recognize the coming back visitors. A simple and straightforward approach in resolving the issue of having users with the same IP or domain name is the elimination of all requests coming from proxies and shared IPs. For example, all requests having the word proxy or cache in their domain name will be eliminated. The drawback of this approach is that navigational patterns of proxy/cache users will not be discovered. Also the dataset may become too small to conduct a valid analysis. In order to mine for navigational patterns it is mandatory to know what visitors have looked at each time they have visited the website. Each time a visitor comes to the website is considered a session. Identifying users' sessions from the web log is not easy as it may seem. Logs may span long period of time during which visitors may come to the website more than once. Therefore, sessions identification becomes the task of dividing the sequence of all page requests made by the same user during that period into subsequences called sessions. Many approaches have been used by researchers for sessions' identification. According to, the most popular session identification techniques use a time gap between requests.

GSP mines sequential patterns based on an Apriori like approach by generating all candidate sequences. This is inefficient and ineffective.

In SPADE, 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.

The database projection growth based approach, FreeSpan, 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, 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. Here each projected database, sequential patterns are grown by exploring only local frequent patterns.

V. PROPOSED WORK

In this paper, we using a novel approach Self Organizing Map (SOM), which is a type of neural network. In the process of Web Usage Mining [13] to detect user's patterns it is usage as a trend analysis. It depends on the performance of the clustering of the quantity of requests. Here we are using SOM algorithm with STPMW (Sequential Traversal Pattern Mining with Weight Constraint) algorithm. The procedure details the transformations essential to modify the data storage with clustered in the Web Servers Log files to an input of SOM. By proceeding this way, first we use SOM algorithm and getting some cluster of web-logs. Here we load that web-log cluster which is almost related to frequent pattern. After that we are applying min weight-max weight of Page in Sequential Traversal Pattern. Finally we establish good prediction with quantity of results.

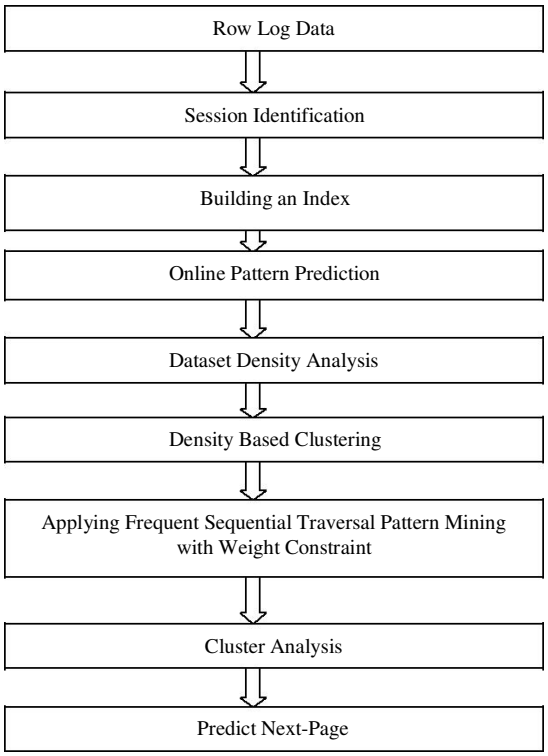

Figure-1 : The above diagram shows the process of proposed work. The step of session identification process go through a density based clustering to mine for useful knowledge. In the same time they go through an efficient online kNN to find the sessions that are mostly related to the current online session.

A. SOM Algorithm for getting closed related data by clustering –

Algorithm 1 (SOM: Self Organizing Map)

Select output layer network topology
– Initialize current neighborhood distance, D (0), to a affirmative value.
• Initialize weights from input to output to small random values. Let t = 1
• As computational bounds are not exceeded do
1) Select an input sample i1
2) Compute the square of the Euclidean distance of i1 from weight vectors (wj) associated with each output node

Σnk=1 (il,k - wj,k (t))2

3) Select output node j* that has weight vector with minimum value from step 2)

4) Update weights to every nodes within a topological distance given by D(t) from j*, using the weight update rule: Wj(t+1) = wj(t) + η (t) (i1 – wj(t))

5) Increment
t Endwhile
Learning rate generally decreases with time
: 0 <η (t) ≤η (t −1) ≤1

Step by step SOM description :

1. Initialize the centroids.
2. Repeat
3. Select the next object.
4. Determine the closest centroid to the object.
5. Update this centroid and the centroids that are close, i.e., in a specified neighborhood until the centroids don't change much or a threshold is exceeded.
6. Assign each object to its closest centroid and return the centroids and clusters.

B. STPMW Algorithm

The divide-and-conquer strategy is used for finding frequent sequential traversal patterns. To handle the ordered problem, the STPMW uses a merging technique. Every frequent ordered pattern whose initial page is P1 must be contained in one or more than one session. The merging procedure in fact is rebuilding a smaller FSTP-tree. This time, the relative sessions all contains P1 as the initial web page.

The complete algorithm given as:
Algorithm 2 (STPMW: Sequential traversal pattern mining with weight constraint)
Input: FSTP-tree
Output: FSTP (frequent sequential traversal pattern)
Method: call STPMW(support, min & max weight of every page)

```
Procedure STPMW (FSTPtree RootNode node, String prefix)
{
  for every node x in the corresponding page head table do
   if x.support less than min support then
      calculate the average weight of prefix
    if min weight<=average weight<=max weight
    {
      output prefix; }
return;
else if i.subs.count == 0 then prefix = prefix + i.content;
calculate the average weight of prefix
if min. weight<=average weight<=max. weight
{
      output prefix;
}
      Update_Cluster_Index(); // using SOM algorithm
return;
else
call CombineTree(i);
for every node j in i.subs do.
call FSTP(j, prefix+i);
end for
end if end for
}
```

Procedure:

Step 1- First we apply SOM algorithm and by this algorithm we are getting some cluster of web-logs according to similarity.

Step 2- Then we load that web-log cluster, which is almost related to FSTP.

Step 3- After that we apply min-max weight of Page in Sequential Traversal Pattern.

Step 4- Finally we establish good cluster items for prediction into the caching to improve the quality of the results and response time..

The following table-1 showing page details with Support and Min-Max weight range.

S.No.	Page ID	Page Name	Support	Min. Weight	Max. Weight
1	P1	Books	9	2	31
2	P2	Electronics	7	3	7
3	P3	Cloths	7	4	22
4	P4	Jewellary	6	5	9
5	P5	Furniture	6	3	10
6	P6	Toys	1	1	2
7	P7	Root	2	1	3

Table-1. "The example of page with weight range".

The following Table-2 showing the Items details of every page which belongs to page.

PageItemId	PageId	Item in Page
1	1	Item-1
2	1	Item-3
3	1	Item-4
4	2	Item-1
5	2	Item-1
6	2	Item-2
7	3	Item-1
8	3	Item-2
9	3	Item-1
10	4	Item-1
11	4	Item-1
12	4	Item-4
13	5	Item-1
14	5	Item-4
15	5	Item-2
16	6	Item-1
17	6	Item-1
18	6	Item-3

Table-2. "The example of page with item".

The following Table-3 showing the Running time (in ms) when we having different database record size with different supports.

Size	Supp-1	Supp-2	Supp-3	Supp-4	Supp-5	Supp-6
1033	6817	6708	6723	6505	6474	6708
2040	18408	18111	18252	18158	18376	18470

3050	15865	17953	18565	18764	18451	18487
4010	24310	24688	24927	25205	26365	24949
5030	83413	38079	84630	54116	40731	38391

Table-3 : Running Time (in ms) with different size and different support

The following Figure-2 showing the Running time (in ms) when we having different record size with different support.

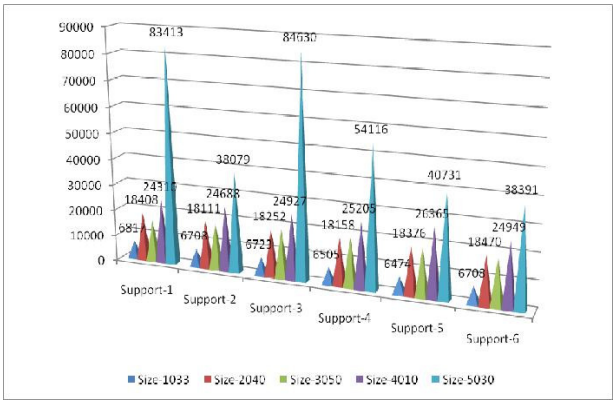

Figure-2 : Running Time (in ms) with different size and different support

The following Table-4 showing the probability of occurrence of each item with different support.

	Item-1	Item-2	Item-3	Item-4
Support-3	0.44	0.19	0.15	0.22
Support-6	0.50	0.17	0.08	0.25
Support-10	0.43	0.20	0.13	0.23
Support-20	0.33	0.00	0.33	0.33

Table-4 : Probability of Items with different support

The following Figure-3 showing the probability of occurrence of each item with different support.

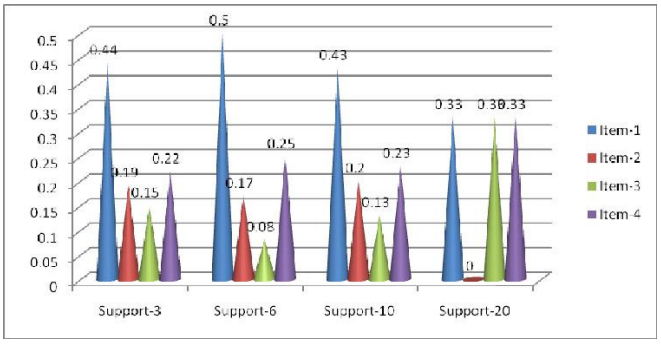

Figure-3 : Probability of items with different support

The following Table-5 showing Info-Gain of Item with different support.

	Item-1	Item-2	Item-3	Item-4
Support-3	07.02	06.08	05.51	06.51
Support-6	12.00	10.34	07.17	12.00
Support-10	07.84	06.97	05.81	07.35
Support-20	01.00	01.00	00.00	01.00

Table-5 : Info-Gain of items with different support

The following Figure-4 showing Info-Gain of Item with different support.

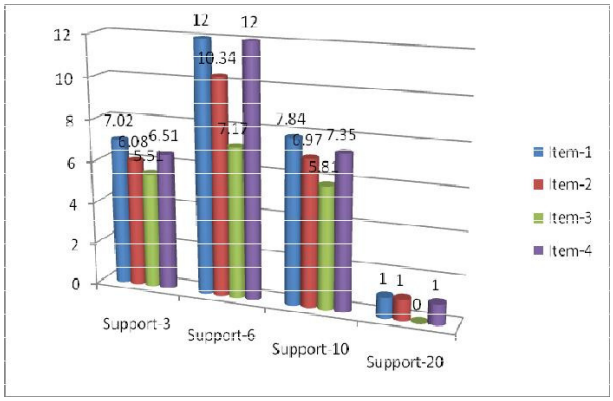

Figure-4 : Info-Gain of items with different support

The following Table-6 showing the comparison between WSpan and STPMW Algorithm with different support. Here we using record size 5030 in the database.

Support	**1**	**2**	**3**	**4**	**5**	**6**
Wspan	83413	83210	80745	83397	84645	83990
STPMW	82009	83100	84630	54116	40731	38391

Table-6 : Comparison of WSpan and STPMW Algorithm with different support (By using Record-Size 5030)

The following Figure-5 showing comparison between WSpan and STPMW algorithm.

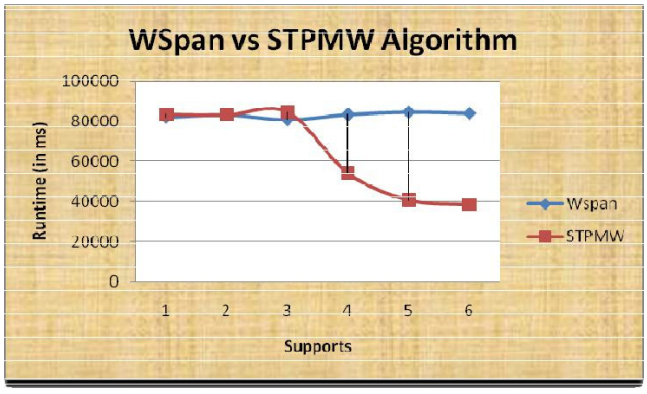

Figure-5 : Comparison of WSpan and STPMW Algorithm with different support

In the above figure-5 showing the comparison between WSpan and STPMW algorithm. If the support either 1 or 6 the execution time of STPMW algorithm is less. So our STPMW algorithm is more efficient.

VI. ANALYSIS AND PERFORMANCE EVALUATION

In this section, we present our performance study over various datasets (eg. 1000, 2000, 3000, 4000 and 5000 sessions) and also with different support (eg. 3, 6, 10 and 20).

We report our experimental results on the performance of STPMW in comparison with a recently developed algorithm, WSpan [1], 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 min-max weight constraint can be generated by incorporating a support and weight page with clustering. First, we show how the number of sequential traversal patterns can be adjusted through user allocate weights, the efficiency in terms of runtime of the STPMW algorithm, and the quality of sequential traversal patterns. Second, we show that STPMW has put related items in cache. Third we are using web services which provide automatically update min-max weight of every page in every fifteen (15) days. It is also decrease back and forth time while finding next page from cache because it also store related page prior in cache..

VII. FURTHER EXTENSION

STPMW algorithm basically focuses on sequential pattern mining with average weight constraint uses a weight range to adjust the number of sequential traversal patterns with the clustering of session. STPMW can be extended by considering levels of support and/or weight of sequential traversal patterns with number of clustering. We can also extended by using Distributed WebLog. There are many areas just like parallel sequential pattern, grouping of similar type of users, in distributed servers.

VIII. CONCLUSION

In our research just begin to touch on the possibilities of SOMs with the frequent sequential pattern mining. One of the main limitations of the traditional approach for mining sequential traversal patterns is that weight of every page is updated manually but here we updated automatically using web services. Second fully database is scan is done while find the next item. Here we clustered the items so that clustered items are only scan not whole database. Third we use min-max weight and support of every page so that every page having different importance. So It is powerful enough to perform extremely computationally expensive operations in a relatively short amount of time for finding next page prediction.

REFERENCES

[1] R. Moriwal, V. Prakash (2013). An efficient Algorithm for finding frequent Sequential traversal Patterns from Web Logs Based on Dynamic Weight Constraint, *Proceedings of the Third International Conference on trends in Information, Telecommunication and Computing*. vol. 150, (Springer Science + Business Media New York 2013.

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

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

[4] A. Guerbas et al.(2013), Effective web log mining and online navigational pattern prediction, Knowl. Based Syst.(2013), http://dx.doi.org/10.1016/j.knosys.2013.04.014.

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

[6] 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.

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

[8] Shyam M. Guthikonda, "Kohonen Self Organizing Maps, Dec 2005.

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

[10] Mishra et al., Web Usage Mining Using Self Organized Map, International Journal of Advanced Research in Computer Science and Software Engineering Vol. 3, Issue(6), June - 2013, pp. 532-539.

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

[12] Parag Pande, Ravindra Gupta (2010), An Efficient Algorithm For Finding Users Behavior by Weight Alignment of Sequential Traversal Patterns, International Journal of Computer, Information Technology & Bioinformatics (IJCITB) ISSN:2278 7593, Volume-1, Issue-2.

[13] Julija pragarauskait , (vilnius, 2013) , frequent pattern analysis for decision making in big data Doctoral Dissertation Physical Sciences, Informatics (09 P) prepared at Institute of Mathematics and Informatics of Vilnius University in 2008 – 2013.

[14] P. Britos, D. Martinelli, H. Merlino, R. García Martínez, Web Usage Mining Using Self Organized Map, PhD Computer Science Program, of La Plata National University. Software & Knowledge Engineering Center, Buenos Aires Institute of Technology, Intelligent Systems Laboratory, University of Buenos Aires , Argentina, 2007.