Web Usage Mining for Comparing User Access Behaviour using Sequential Pattern

Amit Dipchandji Kasliwal[#], Dr. Girish S. Katkar[*]

[#]*Malegaon, Nashik, Maharashtra, India*

[*]*Dept. of Computer Science, Arts, Sci., Commerce College,*
Koradi, Nagpur, Maharashtra, India

Abstract - **In the current era of technology, every organization using a dynamic process of communication that changing in very interactive manner which is fulfilled by their own website hosted by their own web servers or commercial web server. This website gathers information about user access at every time when user interacts for communication with some resources available to them. And this makes easiest way for website administrator to compare the information about the user's navigational patterns explored from the web access logs and can also be useful to compare the user access behaviour about what will be the next page that should be provided to the user following the patterns. To achieve this purpose sequential pattern technique can be used. Sequential pattern technique discovers frequent sequence referred to as patterns in sequential database. In this study we are proposing the algorithm for finding the sequential pattern that could suggest and compare the generated patterns with the minimum support measure.**

Keywords— **Web Usage Mining, Web access log file, Sequential patterns.**

I. Introduction

Sequential pattern technique discovers frequent sequence referred to as patterns in sequential database. The sequential pattern mining process was introduced by Agrawal et al and Srikant et al [1] based on their study regarding the customer purchase sequences. The problem they studied under process was to find all frequent sequences such that sequences in which items occurs frequently in the set of sequences is not less than the minimum support.

An access made to each webpage of website, this access is get recorded into web access log file. Web log file can be considered as a sequence of web pages maintained in the form of log entry accessed by users. As the main intension of using web usage mining is to discover interesting and frequent user access patterns from the web access log data as shown by Pei et al [9]. Web access log stores the information that can be important for website administrator while improving web sites by link to link access by Srivastava et al [1]. In this chapter the authors are proposing and implementing SequentialPattern algorithm in WebUMining model for comparing user access behaviour. The goal of this chapter is to find sequential patterns from web log file that occurs frequently with respect to minimum support count provided by the user. As, it is an important data mining task and has broad applications, the proposed algorithm is implemented to provide the requested user with more useful and interesting patterns. A web access by user is a pattern that is pursued frequently by users. Using these sequences as starts, researcher is converting it into the sequential database so that it can be mine with proposed algorithm. The proposed algorithm mines the complete set of patterns but greatly reduces the efforts of candidate generated through subsequence from patterns. Next on, the proposed work substantially reduces the size of irrelevant sequential patterns and hence leads to efficient processing.

Sequential pattern mining is an important data mining technique can be used for retrieving patterns from large size of data. It is also used to mine frequent navigational paths among user accesses. Web Access Pattern is a sequential pattern technique in a large collection web log data. A web log record can be referred as the sequence of pairs containing user's IP id and requested URL. Sequential pattern mining is an efficient technology used for extracting access patterns. For the sequential pattern process, let us consider itemset $I_s = (i_1, i_2, \ldots, i_n)$ and X is the subset such that $X\ C\ I_s$. Let S is the sequence in ordered manner list of itemset denoted by $(S_1S_2 \ldots S_i)$, where S_i is itemset of the sequence denoted by $(X_1X_2 \ \ldots \ X_m)$ where X_m is the item in sequence S_i. If an item occurs only one item then it can be written as X. The number of times the item occurs in a sequence is called the length of the sequence. A sequence is then get store into database D as a set of element as $\{S_{id}, S\}$, where S_{id} is sequence identifier and S is sequence. The support count for a sequence α in sequential database D, S is the number of tuples in the database containing α, given as $SUPPORT_S\ (\alpha) = |\ \{S_{id}, S\} \ni (S, \alpha \in D\ )\ |$. For the user defined support count, a sequence α is said to be called a sequential pattern in sequential database S if and only if $SUPPORT_S(\alpha) \geq Min\_Support\_Count$. To demonstrate the web access pattern through web log access data, let us consider set of user access during a particular session denoted by E used to represent web pages accessed by users. A web access sequence is ordered sequence that gives the ordered list web pages accessed by user in a session. It is given by S as $S = \{(E_1, E_2, \ldots, E_n)\ |\ E_i \ni E \text{ and } 1 < i < n\ \}$ and in the sequential pattern the E_i and E_j are not necessary to be different.

Sequential Pattern Mining Algorithm is basically based on two approaches apriori based and pattern growth based approach. It mines frequent patterns and sequential pattern containing frequent itemset which are treated uniformly in the training set. For example, in sequential pattern mining, a sequential pattern {(computer, software), (software, antivirus)} can be discovered with user defined support count as it is relatively high, almost in all scenario. And that makes it's limitation for the real life itemset. If we consider the real life scenario, items have important sequences such as {(event, charity), (car, insurance), (computer, webcam)} could not be mine with the traditional sequential pattern mining approach because the itemset contains item that may have low support count. And hence, in real life scenario items with less (or low) support count become more important due to some features of the item may have. After understanding this scenario, the proposed SequentialPattern algorithm is introduced by implementing the weight based pattern mining algorithm again with the modification in the main algorithm suggested by Chen et al, Huang et al [11]. In traditional sequential pattern mining technique, all sequences basically considered with same importance, but as in real life examples, sequences varies in their importance.

The proposed algorithm implemented here is with the key concept that to assign some weightage to the items according to their importance as they occurs in the sequence. In the consequence to the word weightage, the word weight can be defined as the value assigned to each item from itemset with respect to their importance. A pair ***(X_i, W_i)*** is called a weighted item where ***X_i*** Є ***X*** is an item and ***W_i Є W*** is the weightage associated to ***X_i*** where ***W_i*** is a set of positive integer assigned to an item and ***$X=(X_1, X_2,..,X_n)$*** be the set of unique items. A record is a set of weighted items in which a single item may appear in multiple record with different weightage associated with it. In this way the weightage is assigned to the items that belongs to the frequent item set prepared either by apriori algorithm or pattern growth approach.

II. Proposed Sequentialpattern Algorithm

In Sequential Pattern technique for mining, there are two module in which the first is to assign the weightage to the each frequent item and then discovering sequential pattern. In the proposed SequentialPattern algorithm, the researcher implementing these two parts where in first part, the occurrence of user accesses to webpage is used as to assign the weightage to each item from an access sequence. In the second part of the proposed algorithm, pattern discovering algorithm considers the weight of sequence and used it for finding out the support determined for user session and depending upon this the sequential patterns are discovered. The proposed SequentialPattern algorithm uses modified form of the structure used in WAPTree algorithm proposed by Pei et al[9] to deal with user access. In the tree generated with WAPTree algorithm, nodes of the tree are modified to handle the weight assigned to the items as the items are now the webpages. In the second part of the proposed SequentialPattern algorithm, the researcher uses the data structures and modified it as proposed in FOLMine algorithm by Rajimol et al [7]. These data structures used as per the data available in the form of web access log data for the proposed algorithm as listed below:

Modified Data Structure	**Description**
Item_List (ItLt)	Linked list modified for containing the items and their occurrence in the web access log data.
Curr_Item_List (CrItLt)	Linked list modified so that it can be used for storing the items from the current user access webpages.
First_Occ_List (FsOcLt)	Linked list modified for storing the first occurrences of a given item in the web access log data.
Acc_List (AcLt)	Linked list modified for storing each web access sequence.
Header_List[]	Array structure containing start address of list containing item with their weight represented as Header_List.

Table 1: Data Structures used in the proposed algorithm

III. Pseudo Code And Flowchart

The proposed algorithm at first scans each web access log entry from web access log one after and then stores it with assigning weightage of each item to 0. While reading the items from itemset, the Curr_Item_List get is loaded with items those are present in the access sequence and their occurrences. After completing the scanning of access sequence from web access log, the Item_List is get updated using Curr_Item_List. Perhaps, the weightage of each item from Curr_Item_List is calculated by performing division between the occurrence and total length of the access sequence and weightage in the Acc_List is get updated. After this preprocessing get over, the frequent itemset of weighted items is generated using Item_List. If the occurrence for an item is goes more than support, it is considered as the perfect sequential pattern for which website administrator is looking for.

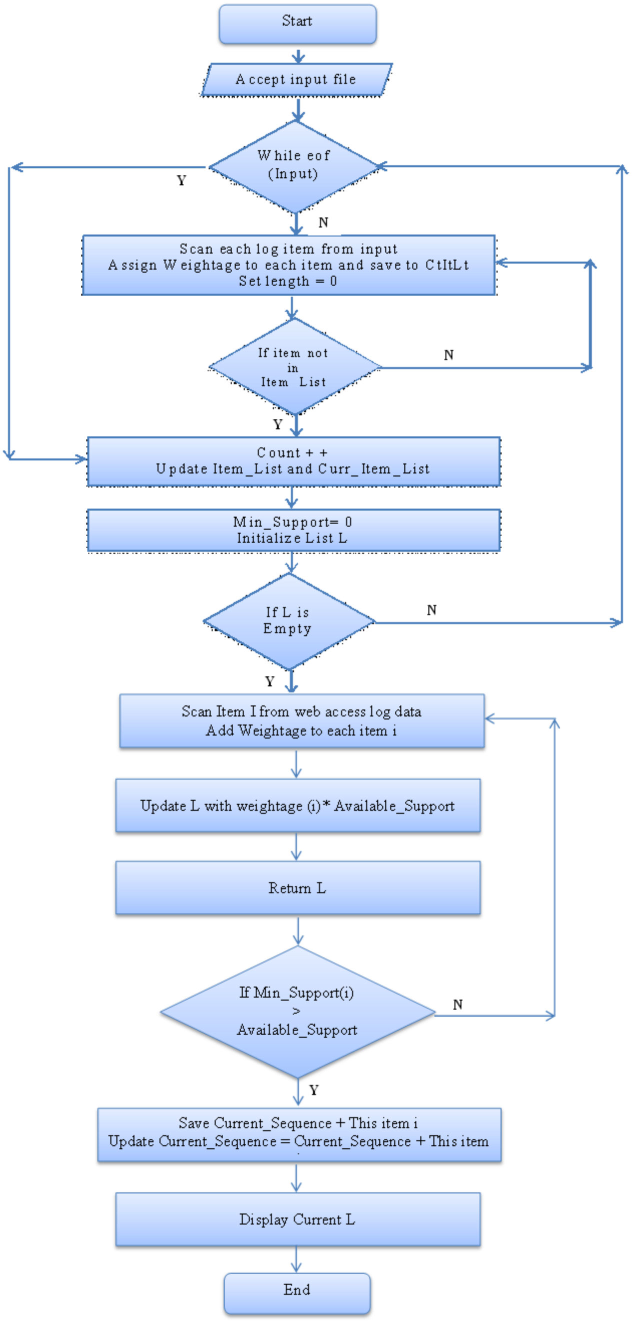
As the data collection and data preprocessing task is already performed and discussed in chapter 2 hence researcher is starting the algorithm with step where WebUMining model is accepting the input file from user. This proposed algorithm is provided with the following pseudo code hence the researcher is intends to have the implementation of the proposed algorithm in the low level prototype model.

```
Algorithm Name: SequentialPattern
Input: Sequential data from Web Access Log,
Minimum Support,
Minimum Confidence.
Output: Sequential Pattern with Support and Confidence.
1. Begin
2. While eof (SequentialWebAccessLogFile)
   Read each web access log AND each item i=0
   Set length = 0
   For each item i in S
           Do length + +
                   If (item i is not in Curr_Item_List)
                         make entry in Curr_Item_List for this i
```

```
                with   count($i$) = $1$
            Else
                count ($i$) ++
            End If
    End For
    Update the Item_List with the Curr_Item_List
    For each $i$
        Update weight ($i$) = count ($i$ ) / length
    End While
3.  For weighted item, $i$
    Set Min_Support = 0 and initialize $List_1$
4.  If $L_1$ is empty
        Then find first occurrence of $i$
    Else If
        GOTO step 2 to locate the first occurrences of
        the element $i$
    End If
5.  Add weightage to item for each occurrence
6.  Update $L_1$ with total weightage * Min_Support
7.  Return $L_1$
8.  If (Min_Support($i$) > Available_Support)
        Then Current Pattern + This Item $i$
        Save Current Pattern + This Item $i$ to Sequences
        Current Pattern = Current Pattern + This Item $i$
        GOTO step 5
    End If
9.  Display current $L_1$
10. End
```

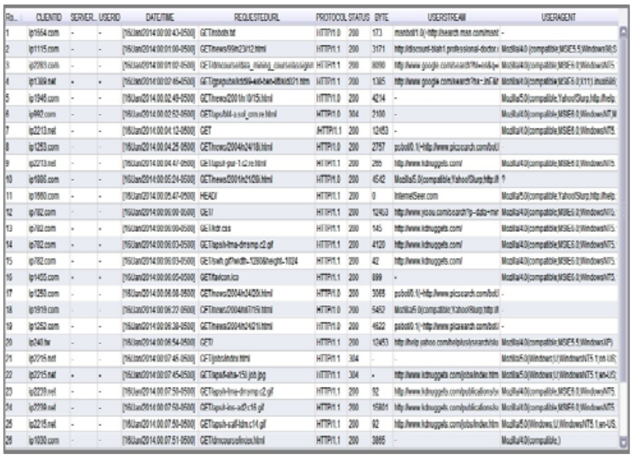
While implementing the proposed SequentialPattern algorithm, initially there was no method to follow a particular session for target page. And hence, the last page in session is taken as the target page. As each webpage refreshed, causes the duplication of webpages in the sequence, therefore nearby duplicate requests from single webpage are eliminated. The output of the proposed SequentialPattern algorithm is in the form of a text file containing the list of sequential patterns with their occurrences as well as support measure and confidence measure. In this format the results are then be easily represented by visualization software. Effectiveness of the proposed SequentialPattern algorithm is implemented and evaluated through comparing processing time and number of patterns generated during the web access log data submitted as as input to algorithm.

Following is the flowchart of the proposed SequentialPattern algorithm as discussed above. The flowchart contains data structures as discussed. The flowchart gives the idea about the flow of data and the actual process implemented in the proposed SequentialPattern algorithm. The algorithm is then implemented with JAVA and it is one of the algorithms of WebUMining model used for web usage mining for comparing ing user access behaviour.


Start
Accept input file
While eof (Input)
Y
N
Scan each log item from input
Assign Weightage to each item and save to CtItLt
Set length = 0
If item not in Item List
N
Y
Count + +
Update Item_List and Curr_Item_List
Min_Support= 0
Initialize List L
If L is Empty
N
Y
Scan Item I from web access log data
Add Weightage to each item i
Update L with weightage (i)* Available_Support
Return L
If Min_Support(i) > Available_Support
N
Y
Save Current_Sequence + This item i
Update Current_Sequence = Current_Sequence + This item
Display Current L
End

IV. EXPERIMENT

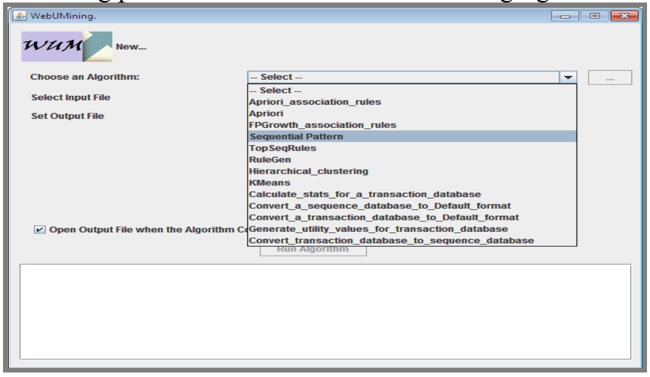
In this section, authors worked on the web access log data to perform the experiment to evaluate the sequential pattern that can be used for comparing user access behaviour through the proposed algorithm implemented in WebUMining model. During experiment, the focus is on fining sequential patterns from web log data, then looking for the presence of the weighted sequential pattern. For conducting the experiments researcher used the web log access data collected from kdnugget's repository named 'kddaccess.arff' which is preprocessed with the DataCleaningPreprocessing algorithm as discussed in chapter 4. Then this file is converted from text file to arff file format using WEKA as shown in figure 1 given below:



Ro..	CLIENTID	SERVER..	USERID	DATETIME	REQUESTEDURL	PROTOCOL	STATUS	BYTE	USERSTREAM	USERAGENT
1	ip1664.com	-	-	[16/Jan/2014:00:00:43-0500]	GET/robots.txt	HTTP/1.0	200	173	msnbot/1.0(+http://search.msn.com/msnbot	-
2	ip1115.com	-	-	[16/Jan/2014:00:01:00-0500]	GET/news/99/n23/12.html	HTTP/1.1	200	3171	http://discount-blah1.professional-doctor.	Mozilla/4.0 (compatible;MSIE5.5;Windows98;S
3	ip2283.com	-	-	[16/Jan/2014:00:01:02-0500]	GET/dmcourse/data_mining_course/assignm	HTTP/1.1	200	8090	http://www.google.com/search?hl=en&q=	Mozilla/4.0(compatible;MSIE6.0;WindowsNT5.
4	ip1389.net	-	-	[16/Jan/2014:00:01:46-0500]	GET/gpspubs/kdd99-web-ban-dt/kdd01.htm	HTTP/1.1	200	1365	http://www.google.com/search?hl=en&ie	Mozilla/4.0(compatible;MSIE6.0;X11) linux686;
5	ip1946.com	-	-	[16/Jan/2014:00:02:49-0500]	GET/news/2001/n10/15i.html	HTTP/1.0	200	4214	-	Mozilla/5.0(compatible;Yahoo!Slurp;http://help.
6	ip992.com	-	-	[16/Jan/2014:00:02:52-0500]	GET/aps/bt4-a.sol_crm.re.html	HTTP/1.0	304	2100	-	Mozilla/4.0(compatible;MSIE6.0;WindowsNT;M
7	ip2213.net	-	-	[16/Jan/2014:00:04:12-0500]	GET	HTTP/1.1	200	12453	-	Mozilla/4.0(compatible;MSIE6.0;WindowsNT5.
8	ip1253.com	-	-	[16/Jan/2014:00:04:25-0500]	GET/news/2004/n24/18i.html	HTTP/1.0	200	2757	psbot/0.1(+http://www.picsearch.com/bot/	-
9	ip2213.net	-	-	[16/Jan/2014:00:04:47-0500]	GET/aps/r-pur-1.c2.re.html	HTTP/1.1	200	285	http://www.kdnuggets.com/	Mozilla/4.0(compatible;MSIE6.0;WindowsNT5.
10	ip1886.com	-	-	[16/Jan/2014:00:05:24-0500]	GET/news/2001/n21/28i.html	HTTP/1.0	200	4542	Mozilla/5.0(compatible;Yahoo!Slurp;http://	?
11	ip1660.com	-	-	[16/Jan/2014:00:05:47-0500]	HEAD/	HTTP/1.1	200	0	InternetSeer.com	Mozilla/5.0(compatible;Yahoo!Slurp;http://help.
12	ip782.com	-	-	[16/Jan/2014:00:06:00-0500]	GET/	HTTP/1.1	200	12453	http://www.ysou.com/search?p=data+min	Mozilla/4.0(compatible;MSIE6.0;WindowsNT5.
13	ip782.com	-	-	[16/Jan/2014:00:06:00-0500]	GET/kdr.css	HTTP/1.1	200	145	http://www.kdnuggets.com/	Mozilla/4.0(compatible;MSIE6.0;WindowsNT5.
14	ip782.com	-	-	[16/Jan/2014:00:06:03-0500]	GET/aps/x-tma-dmsmp.c2.gif	HTTP/1.1	200	4120	http://www.kdnuggets.com/	Mozilla/4.0(compatible;MSIE6.0;WindowsNT5.
15	ip782.com	-	-	[16/Jan/2014:00:06:03-0500]	GET/swh.gif?width=1280&height=1024	HTTP/1.1	200	42	http://www.kdnuggets.com/	Mozilla/4.0(compatible;MSIE6.0;WindowsNT5.
16	ip1455.com	-	-	[16/Jan/2014:00:06:05-0500]	GET/favicon.ico	HTTP/1.1	200	899	-	Mozilla/4.0(compatible;MSIE6.0;WindowsNT5.
17	ip1250.com	-	-	[16/Jan/2014:00:06:08-0500]	GET/news/2004/n24/20i.html	HTTP/1.0	200	3065	psbot/0.1(+http://www.picsearch.com/bot/	-
18	ip1919.com	-	-	[16/Jan/2014:00:06:27-0500]	GET/news/2004/n07/15i.html	HTTP/1.0	200	5457	Mozilla/5.0(compatible;Yahoo!Slurp;http://	-
19	ip1252.com	-	-	[16/Jan/2014:00:06:38-0500]	GET/news/2004/n24/21i.html	HTTP/1.0	200	4622	psbot/0.1(+http://www.picsearch.com/bot/	-
20	ip240.tw	-	-	[16/Jan/2014:00:06:54-0500]	GET/	HTTP/1.1	200	12453	http://help.yahoo.com/help/us/search/slu	Mozilla/4.0(compatible;MSIE5.5;WindowsXP)
21	ip2215.net	-	-	[16/Jan/2014:00:07:45-0500]	GET/jobs/index.html	HTTP/1.1	304	-	-	Mozilla/5.0(Windows;U;WindowsNT5.1;en-US;
22	ip2215.net	-	-	[16/Jan/2014:00:07:45-0500]	GET/aps/l-sha-15l.job.jpg	HTTP/1.1	304	-	http://www.kdnuggets.com/jobs/index.htm	Mozilla/5.0(Windows;U;WindowsNT5.1;en-US;
23	ip2239.net	-	-	[16/Jan/2014:00:07:50-0500]	GET/aps/x-tma-dmsmp.c2.gif	HTTP/1.1	200	92	http://www.kdnuggets.com/publications/s	Mozilla/4.0(compatible;MSIE6.0;WindowsNT5.
24	ip2239.net	-	-	[16/Jan/2014:00:07:50-0500]	GET/aps/r-ins-ad2.c16.gif	HTTP/1.1	200	15801	http://www.kdnuggets.com/publications/s	Mozilla/4.0(compatible;MSIE6.0;WindowsNT5.
25	ip2215.net	-	-	[16/Jan/2014:00:07:50-0500]	GET/aps/x-salf-tdm.c14.gif	HTTP/1.1	200	92	http://www.kdnuggets.com/jobs/index.htm	Mozilla/5.0(Windows;U;WindowsNT5.1;en-US;
26	ip1030.com	-	-	[16/Jan/2014:00:07:51-0500]	GET/dmcourse/index.html	HTTP/1.1	200	3885	-	Mozilla/4.0(compatible;)

Figure 1: Input web access log data file

The WebUMining model accepts this input file and selecting the proposed SequentialPattern algorithm the remaining processes are shown in the following figures.



Choose an Algorithm: -- Select --

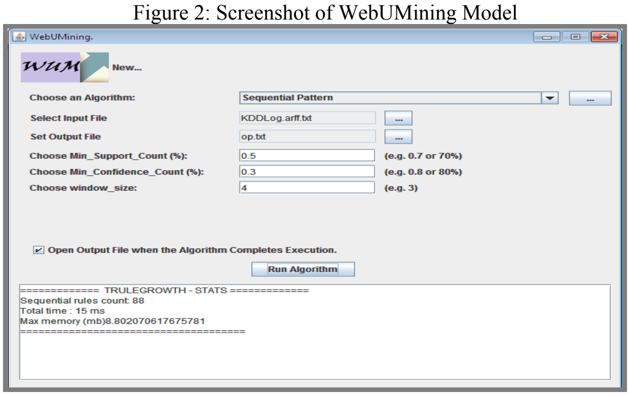
Select Input File

Set Output File

-- Select --
Apriori_association_rules
Apriori
FPGrowth_association_rules
Sequential Pattern
TopSeqRules
RuleGen
Hierarchical_clustering
KMeans
Calculate_stats_for_a_transaction_database
Convert_a_sequence_database_to_Default_format
Convert_a_transaction_database_to_Default_format
Generate_utility_values_for_transaction_database
Convert_transaction_database_to_sequence_database

☑ Open Output File when the Algorithm C

Figure 2: Screenshot of WebUMining Model



Choose an Algorithm: Sequential Pattern

Select Input File: KDDLog.arff.txt

Set Output File: op.txt

Choose Min_Support_Count (%): 0.5 (e.g. 0.7 or 70%)

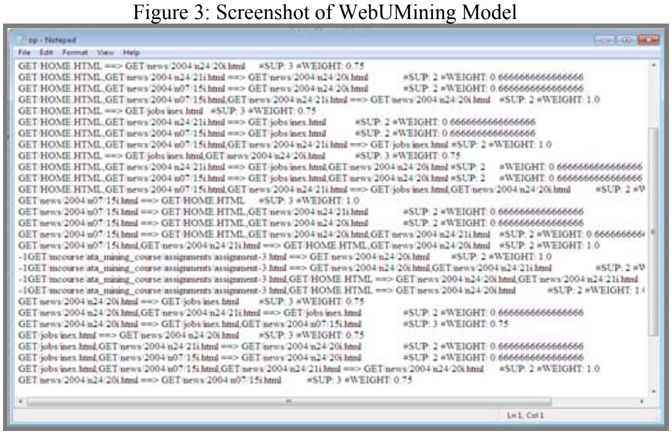
Choose Min_Confidence_Count (%): 0.3 (e.g. 0.8 or 80%)

Choose window_size: 4 (e.g. 3)

☑ Open Output File when the Algorithm Completes Execution.

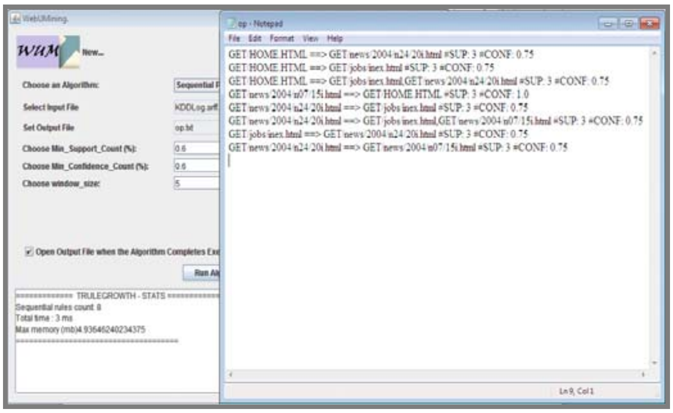
```
============= TRULEGROWTH - STATS =============
Sequential rules count: 88
Total time : 15 ms
Max memory (mb)8.802070617675781
=======================================
```

Figure 3: Screenshot of WebUMining Model



```
GET/HOME.HTML ==> GET/news/2004/n24/20i.html     #SUP: 3 #WEIGHT: 0.75
GET/HOME.HTML,GET/news/2004/n24/21i.html ==> GET/news/2004/n24/20i.html          #SUP: 2 #WEIGHT: 0.6666666666666666
GET/HOME.HTML,GET/news/2004/n07/15i.html ==> GET/news/2004/n24/20i.html          #SUP: 2 #WEIGHT: 0.6666666666666666
GET/HOME.HTML,GET/news/2004/n07/15i.html,GET/news/2004/n24/21i.html ==> GET/news/2004/n24/20i.html   #SUP: 2 #WEIGHT: 1.0
GET/HOME.HTML ==> GET/jobs/inex.html   #SUP: 3 #WEIGHT: 0.75
GET/HOME.HTML,GET/news/2004/n24/21i.html ==> GET/jobs/inex.html         #SUP: 2 #WEIGHT: 0.6666666666666666
GET/HOME.HTML,GET/news/2004/n07/15i.html ==> GET/jobs/inex.html         #SUP: 2 #WEIGHT: 0.6666666666666666
GET/HOME.HTML,GET/news/2004/n07/15i.html,GET/news/2004/n24/21i.html ==> GET/jobs/inex.html #SUP: 2 #WEIGHT: 1.0
GET/HOME.HTML ==> GET/jobs/inex.html,GET/news/2004/n24/20i.html         #SUP: 3 #WEIGHT: 0.75
GET/HOME.HTML,GET/news/2004/n24/21i.html ==> GET/jobs/inex.html,GET/news/2004/n24/20i.html #SUP: 2     #WEIGHT: 0.6666666666666666
GET/HOME.HTML,GET/news/2004/n07/15i.html ==> GET/jobs/inex.html,GET/news/2004/n24/20i.html #SUP: 2     #WEIGHT: 0.6666666666666666
GET/HOME.HTML,GET/news/2004/n07/15i.html,GET/news/2004/n24/21i.html ==> GET/jobs/inex.html,GET/news/2004/n24/20i.html        #SUP: 2 #W
GET/news/2004/n07/15i.html ==> GET/HOME.HTML     #SUP: 3 #WEIGHT: 1.0
GET/news/2004/n07/15i.html ==> GET/HOME.HTML,GET/news/2004/n24/21i.html          #SUP: 2 #WEIGHT: 0.6666666666666666
GET/news/2004/n07/15i.html ==> GET/HOME.HTML,GET/news/2004/n24/20i.html          #SUP: 2 #WEIGHT: 0.6666666666666666
GET/news/2004/n07/15i.html ==> GET/HOME.HTML,GET/news/2004/n24/20i.html,GET/news/2004/n24/21i.html   #SUP: 2 #WEIGHT: 0.666666666
GET/news/2004/n07/15i.html,GET/news/2004/n24/21i.html ==> GET/HOME.HTML,GET/news/2004/n24/20i.html   #SUP: 2 #WEIGHT: 1.0
-1GET/mcourse/ata_mining_course/assignments/assignment-3.html ==> GET/news/2004/n24/20i.html        #SUP: 2 #WEIGHT: 1.0
-1GET/mcourse/ata_mining_course/assignments/assignment-3.html ==> GET/news/2004/n24/20i.html,GET/news/2004/n24/21i.html        #SUP: 2 #W
-1GET/mcourse/ata_mining_course/assignments/assignment-3.html,GET/HOME.HTML ==> GET/news/2004/n24/20i.html,GET/news/2004/n24/21i.html
-1GET/mcourse/ata_mining_course/assignments/assignment-3.html,GET/HOME.HTML ==> GET/news/2004/n24/20i.html         #SUP: 2 #WEIGHT: 1.
GET/news/2004/n24/20i.html ==> GET/jobs/inex.html       #SUP: 3 #WEIGHT: 0.75
GET/news/2004/n24/20i.html,GET/news/2004/n24/21i.html ==> GET/jobs/inex.html            #SUP: 2 #WEIGHT: 0.6666666666666666
GET/news/2004/n24/20i.html ==> GET/jobs/inex.html,GET/news/2004/n07/15i.html            #SUP: 3 #WEIGHT: 0.75
GET/jobs/inex.html ==> GET/news/2004/n24/20i.html       #SUP: 3 #WEIGHT: 0.75
GET/jobs/inex.html,GET/news/2004/n24/21i.html ==> GET/news/2004/n24/20i.html            #SUP: 2 #WEIGHT: 0.6666666666666666
GET/jobs/inex.html,GET/news/2004/n07/15i.html ==> GET/news/2004/n24/20i.html            #SUP: 2 #WEIGHT: 0.6666666666666666
GET/jobs/inex.html,GET/news/2004/n07/15i.html,GET/news/2004/n24/21i.html ==> GET/news/2004/n24/20i.html     #SUP: 2 #WEIGHT: 1.0
GET/news/2004/n24/20i.html ==> GET/news/2004/n07/15i.html        #SUP: 3 #WEIGHT: 0.75
```

Figure 4: Result Screen of SequentialPattern algorithm



Choose an Algorithm: Sequential P

Select Input File: KDDLog.arff

Set Output File: op.txt

Choose Min_Support_Count (%): 0.6

Choose Min_Confidence_Count (%): 0.6

Choose window_size: 5

☑ Open Output File when the Algorithm Completes Exe

```
============= TRULEGROWTH - STATS =============
Sequential rules count: 8
Total time : 3 ms
Max memory (mb)4.93646240234375
=======================================
```

```
GET/HOME.HTML ==> GET/news/2004/n24/20i.html #SUP: 3 #CONF: 0.75
GET/HOME.HTML ==> GET/jobs/inex.html #SUP: 3 #CONF: 0.75
GET/HOME.HTML ==> GET/jobs/inex.html,GET/news/2004/n24/20i.html #SUP: 3 #CONF: 0.75
GET/news/2004/n07/15i.html ==> GET/HOME.HTML #SUP: 3 #CONF: 1.0
GET/news/2004/n24/20i.html ==> GET/jobs/inex.html #SUP: 3 #CONF: 0.75
GET/news/2004/n24/20i.html ==> GET/jobs/inex.html,GET/news/2004/n07/15i.html #SUP: 3 #CONF: 0.75
GET/jobs/inex.html ==> GET/news/2004/n24/20i.html #SUP: 3 #CONF: 0.75
GET/news/2004/n24/20i.html ==> GET/news/2004/n07/15i.html #SUP: 3 #CONF: 0.75
```

Figure 4: Result screen with different support value

V. VISUALIZING SEQUENTIALPATTERN RESULT

The visualization is the vital part of the research work. It is used to compare the user access behaviour when processing web access log data through data mining techniques for achieving web usage mining process. When visualizing user access sequential pattern it is important to present the structure of the web site. Basically sequential pattern used to compare the start and end points that web user used during surfing the internet. It not only shows which pages accessed by users but also gives the order in which they are visited. Figure 5.6 shows the all sequential pattern that ate found in the sample data collected from the kdnugget's repository.

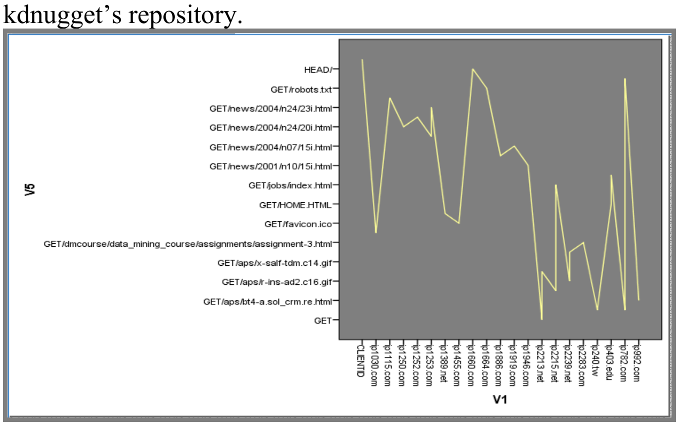
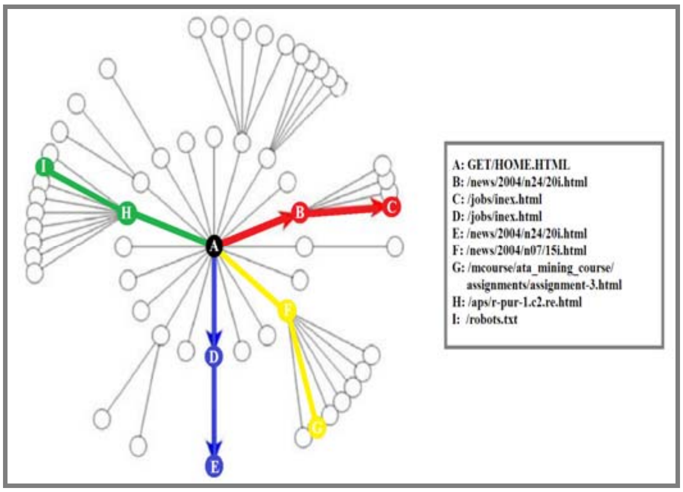


Figure 6: Visualization of Sequential Pattern



A
B
C
D
E
F
G
H
I
A: GET/HOME.HTML
B: /news/2004/n24/20i.html
C: /jobs/inex.html
D: /jobs/inex.html
E: /news/2004/n24/20i.html
F: /news/2004/n07/15i.html
G: /mcourse/ata_mining_course/assignments/assignment-3.html
H: /aps/r-pur-1.c2.re.html
I: /robots.txt

Figure 7: Comparison of Sequential Pattern

No. of Sequential Pattern Generated	Time (in MSec.)	Minimum Support
248	624	0.2
56	156	0.3
56	44	0.4
44	41	0.5
11	8	0.6

Table 3: Comparison of result generated by proposed SequentialPattern Algorithm

The table given above shows the sequential pattern generated by proposed SequentialPattern algorithm with respect to the minimum support measure and the time taken to generate the result so that it is useful to compare the sequential pattern with respect to the given support count.

VI Conclusion

This research work proposed by the authors demonstrate a low level prototype called WebUMining which implentemented with the proposed SequentialPattern Algorithm which can be used to compare the user access behaviour using web access log data through web usage mining techniques. The proposed algorithm is implemented using Java and WEKA. For visualization of the result we have used the SPSS and RapidMiner software.

References

[1] Agrawal R. and Srikant R., “Mining sequential patterns”, International Conference on Data Engineering , 1995.

[2] Yun U., “A New Framework for Detecting Weighted Sequential Patterns”, Large Sequence Databases. Knowledge-Based Systems, 2008.

[3] Masseglia F., Teisseire M. and Poncelet P., “Sequential Pattern Mining: A Survey on Issues and Approaches”, Encyclopedia of Data Warehousing and Mining”, 2005.

[4] Gupta M. and Han J., “Approaches for Pattern Discovery Using Sequential Data Mining”, Pattern Discovery using Sequential Data Mining, 2011.

[5] Tyagi N., Tyagi S. and Solanki A., “An Algorithmic approach to data preprocessing in Web usage mining”, International Journal of Information Technology and Knowledge Management, 2010.

[6] Rathod K. and Valera,“ A Novel Approach of Mining Frequent Sequential Pattern from Customized Web Log Preprocessing”, International Journal of Engineering Research and Applications, 2013

[7] Rajimol, “Data Mining For Web Intelligence”, Thesis, MG University, 2013.

[8] Vasumathi, “Web Mining Using Pattern Discovery Techniques”, Thesis, Jawaharlal Nehru Technological University, 2010.

[9] Han J. and Pet J., “Mining Frequent Patterns by Pattern-Growth: Methodology and Implications”, SIGKDD Explorations, 2000.

[10] Cooley R., Mobasher B. and Srivastava J., “Web Mining: Information and Pattern Discovery on the World Wide Web”, International Conference on Tool and Artificial Intelligence, 1997.

[11] Chen y. and Huang T., “Discovering Time-Interval Sequential Patterns in Sequence Databases”, Expert Systems with Applications, 2003.

[12] http://www.kdnugget.com