ISSN:0975-9646
Zahid Ansari et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 2 (4) , 2011, 1407-1413

A Comparative Study of Mining Web Usage Patterns Using Variants of *k*-Means Clustering Algorithm

Zahid Ansari[#], A. Vinaya Babu[*], Waseem Ahmed[#]and Mohammed Fazle Azeem[†,]

[#]*Department of Computer Science Engineering, P.A. College of Engineering,Mangalore, India*
[*] *Department of Computer Science Engineering, J.N.T. University, Hyderabad, India*
[†]*Department of Electronics and Communication Engineering, P.A. College of Engineering,Mangalore, India*

***Abstract*— The explosive growth in the information available on the Web has prompted the need for developing Web personalization systems that understand and exploit user preferences to dynamically serve customized content to individual users [1]. To reveal information about user preferences from Web usage data, Data Mining techniques can be naturally applied, leading to the so-called Web Usage Mining (WUM) [2]. Clustering is widely used in WUM to capture similar interests and trends among users accessing a Web site [3]. k-Means clustering is a popular clustering algorithm based on the partitioning of data. However one of the drawbacks of it is that it requires the user to specify the number of clusters at the beginning and also it is sensitive to the initial selection of cluster centres. The global *k*-Means algorithm proposed by Likas [4] provides an incremental approach to clustering by dynamically adding one cluster centre at a time through a deterministic global search procedure. It does not depend on any initial conditions and considerably outperforms the *k*-Means algorithms, but the problem associated with this algorithm is its heavy computational effort. A faster version of global *k*-Means algorithm substantially reduces the execution time by improving the way of creating the next cluster centre in the global *k*-Means algorithm. We implemented and tested these algorithms against the web usage data in order to discover the user navigational session clusters. In this paper we present the implementation details of each of the above mentioned *k*-Means clustering techniques along with the underlying mathematical foundations. The results are presented with a comparison of different techniques. Our results show that the fast global *k*-Means clustering algorithm significantly reduces the computational time without affecting the performance of the global *k*-Means algorithm. It also outperforms the global K-means algorithm in terms of validity measure.**

***Keywords*— web usage mining, *k*-Means clustering, global *k*-Means clustering, fast global *k*-Means clustering.**

I. INTRODUCTION

Data mining is the process of exploration and analysis, by automatic or semiautomatic means, of large quantities of data in order to discover meaningful patterns or rules. It deals with the "knowledge in the database" [5]. The term KDD refers to the overall process of knowledge discovery in databases. Data mining is a particular step in this process, involving the application of specific algorithms for extracting patterns from data. The additional steps in the KDD process, such as data preparation, data selection, data cleaning, incorporation of appropriate prior knowledge, and proper interpretation of the results of mining, ensures that useful knowledge is derived from the data [6]. The aim of data mining techniques is to discover hidden information from data sets and to present it in an understandable way. The common model functions in current data mining practice include classification, regression clustering, rule generation, discovering association, summarization and sequence analysis [7]. Data mining often builds on an interdisciplinary bundle of specialized techniques from fields such as statistics, artificial intelligence, machine learning, data bases, pattern recognition, computer-based visualization etc.

Web Usage Mining [8] is described as the automatic discovery and analysis of patterns in web logs and associated data collected as a result of user interactions with Web resources on one or more Web sites. The goal of Web usage mining is to capture, model, and analyse the behavioural patterns and profiles of users interacting with a Web site. The discovered patterns are usually represented as collections of URLs that are frequently accessed by groups of users with common interests. Web usage mining has been used in a variety of applications such as i) Web Personalization systems [1], ii) Adaptive Web Sites [9][10], iii) Business Intelligence [11], iv) System Improvement to understand the web traffic behaviour which can be utilized to decide strategies for web caching [12], load balancing and data distribution [13], iv) Fraud detection: detection of unusual accesses to the secured data [14], etc.

In this paper the following variants of K-means clustering techniques are reviewed: i) K-means (or Hard C-means) clustering [15], ii) Global K-means clustering and iii) Fast Global K-means clustering [4]. These techniques are implemented and tested against the web access log data. The results are presented with a comparison of the different techniques and the effect of different parameters in the process.

The remainder of the paper is organized as follows. Section II presents an overview of web usage data preprocessing techniques and the underlying concepts. Section III presents each of the k-Means clustering techniques in detail along with the underlying mathematical foundations. Section IV describes the experimental results of each technique, followed by a comparison of the results. A brief conclusion and future work are presented in Section V.

II. WEB USAGE DATA MODELLING AND PREPROCESSING

In accordance with the W3C's web characterization terminologies, [16] and [17] provided definitions for the main WUM terms which are described in Table I.

TABLE I

DESCRIPTION OF WUM TERMS

WUM Term	Description
Resource	According to the W3C's URI specification, it can be anything that has identity.
URI	A compact string of characters to identify a resource. e.g. an HTML file or an image.
Web resource	A resource accessible through any version of the HTTP protocol.
Web server	The server that provides access to Web resources.
Web page	Set of data constituting one or several Web resources that can be identified by a URI.
Page View	Is a set of files that contribute to the display of a web page on a user's web browser at a specific moment in time.
Web Browser	Is client software that can send Web requests, handle the responses, and display requested URIs.
User	Is an individual that is accessing files from one or more web servers using a Web browser.
Web request	Is a request, a Web client makes for a Web resource.
Clickstream	Is a sequential series of a page view requests
User session	Clickstream of page views for a single user across one or more Web servers.
Visit/Server session	The set of page views in a user session for a particular Web site
Episode	Is a subset of a visit constituted from related clicks.

Web Usage Mining like any knowledge, discovery and data mining process, performs three main steps: preprocessing, pattern extraction and results analysis. The goal of the preprocessing stage in Web usage mining is to transform the raw web log data into a set of user sessions. Each session consists of a sequence or a set of URLs corresponding to pageviews. This sessionized data can be further transformed and abstracted or used as the input for various data mining algorithms. Web usage data preprocessing exploit a variety of algorithms and heuristic techniques for various preprocessing tasks such as data fusion and cleaning, user and session identification [8] etc. The primary tasks in usage data preprocessing are described below.

A. Data Fusion:

Many important organizations have several Web servers for their Web sites. [16] In order to perform analysis of the of the users' behaviours on the entire Web site, web log files generated by different web servers should be combined [18]. Data fusion refers to the merging of log files from several Web servers. This requires global synchronization across these servers. The Web server's name is added in the requests before the file path. Web server clocks are synchronized across various time zones and the original host name is replaced with an identifier [16].

B. Data cleaning:

Data cleaning involves tasks such as, removal of irrelevant references to embedded objects, style files, graphics, or sound files, and references due to spider navigations. Cleaning of a server log to eliminate irrelevant items are of primary importance for any type of Web log analysis. The discovered patterns are only useful if the server log data provides an accurate picture of the user accesses to the Web site. A user's request to view a particular page often results in several log entries since graphics and scripts are down-loaded in addition to the HTML file. In most cases, only the log entry of the HTML file request is relevant because the user does not explicitly request all of the graphics that are on a Web page, they are automatically downloaded due to the HTML tags. Since the main intent of Web Usage Mining is to get a picture of the user's behaviour, it does not make sense to include file requests that the user did not explicitly request. Data cleaning also identifies Web robots and removes their requests. Popular Web sites generate gigabytes of web log data per hour. Processing such huge files is a tedious task. By performing data cleaning, log file sizes can be reduced to enhance the subsequent mining tasks. On the other hand not all page requests are recorded in server access logs. Client-side or proxy-side caching can often result in missing references to those pages or objects that have been cached. In order to infer these missing references, Cooley et. al. have described a process called path completion which relies on the knowledge of site structure and referrer information from server log files [19].

Elimination of image file requests: Most Web pages contain images. Whether to retain or eliminate the log files for these images depends much on the goal of the Web Usage Mining. To support Web caching or pre fetching, the log entries corresponding to images and multimedia files should be retained because predicting requests for such files with is more important than the HTML documents due to their large sizes. In contrast, for the purpose of web personalization or website structure redesign these requests should be eliminated [16]. Elimination of the image and other multimedia requests can be accomplished by checking the suffix of the URL name. For example, log entries with file name suffixes such as, gif, jpeg, GIF, JPEG, jpg, JPG, and map can be discarded. A default list of suffixes can be maintained for this purpose.

Elimination of robots' requests: Web robots or spiders are software tools that automatically traverse hyperlink structure of world wide web in order to locate an retrieve information [20]. Sessions consisting of Web robot request make the effective Web usage analysis more difficult. Web Robots scan all the pages from a Web site to update their search indexes. The number of requests from a Web Robot is at least the number of the site's URIs. Many times for some unpopular web site, the number of requests from all the Web Robots that have scanned that site exceed other user requests. Elimination of log entries corresponding to the Web robot requests, filters non useful sessions and enhances the subsequent mining tasks. In order to identify a Web Robot request following heuristics are being used [16][18]: i) Find all the hosts that have requested the page "robots.txt". ii) Refer a list of all user agents known as robots. iii) Compute the browsing speed of the host, If the browsing speed and the number of pages visited during the current visit exceed some specified threshold, then the host is considered as web robot.

C. User identification:

Next, unique users must be identified. For Web sites requiring user registration, the web logs contain the user login name. In such cases this information can be used for user identification. For those cases where user login information is not available, each IP might be considered as a user. Pirolli et. al. [21] suggests that each different agent type for an IP

address represents a different user. Another approach described in [19] uses the access log along with the referrer log and site topology to construct browsing paths for each user. If a requested page is not directly reachable from any of the pages visited by the user, then this represents another user with the same IP address.

D. User session identification:

User Session identification is the process of segmenting the user activity log of each user into sessions, each representing a single visit to the site. It is a complicated task, due to the presence of proxy servers, dynamic addresses, and cases of multiple users accessing the same computer. It is also possible that one user might be using multiple browsers or computers [16]. For the Web sites without user authentication information, Spiliopoulo et al. [22] describe a proactive strategy for session identification. Web server software is modified to associate a unique identifier to each client process accessing the server. Web log corresponding to each request made by the user's client to the server contains this identifier, which enables the unique assignment of requests to users during one visit. The identifier expires only when the client process is terminated, connection is broken or a timeout occurs. Expiry of the identifier indicates the end of the session. Web sites without user authentication information or embedded session ids mostly rely on heuristics methods for sessionization. The sessionization heuristic helps in extracting the actual sequence of actions performed by one user during one visit to the site. Generally, sessionization heuristics are either time-oriented or structure oriented [19]. A formal framework for measuring the effectiveness of such heuristics has been proposed in [22]. Berendt et. al. investigated the impact of site different heuristics on the quality of constructed user sessions [23]. Time-oriented heuristics consider an upper bound on the time spent in the entire site during a visit or an upper bound on page-stay time [22]. If the time between page requests exceeds a certain threshold, it is assumed that the user is starting a new session. Based on empirical data, Cateledge and Pitkow [25] proposed a timeout of 25.5 minutes. based on empirical data. A default time threshold of 30 minutes is commonly used by many tools [19][24].

A second type of time-oriented heuristic uses a threshold on the total page-stay time. There is no commonly used threshold for page-stay time due to the fact that page-stay time is affected by the page information content, page loading time and by data transfer rate of the communication line.

Navigation-oriented heuristics considers Web navigational behaviours of the [22]. Web users typically reach pages by following hyperlinks rather than by typing URLs. Therefore, if a page request that is unreachable through the pages visited by the user so far is likely to have been initiated by another user. This strategy is used to partition a Web server log into sessions. According to Cooley et. al. [19], a requested Web page P need not be accessible from the page immediately accessed before it. Rather, it might be reachable from a page the user backtracked to. Web server logs may not contain these backward moves because of the client catching mechanism. If a page request is not directly linked to the last requested page and if the referrer of this page is present in the user's recent request history, then the user might have backtracked, receiving cached versions of the pages until a new page was requested. Missing page references are then added to the user session file. They have referred this strategy as "Path Completion". Site topology can also be used as an alternative for the referrer log for the same purpose.

III. CLUSTERING METHODOLOGY

Clustering aims to divide a data set into groups or clusters where inter-cluster similarities are minimized while the intra cluster similarities are maximized. Details of various clustering techniques can be found in survey articles [26]-[28].

In order to perform clustering on user session data, we map the user sessions as vectors of URL references in a *n*-dimensional space. Let $U = \{u_1, u_2, \ldots, u_n\}$ be a set of *n* unique URLs appearing in the preprocessed log and let $S = \{s_1, s_2, \ldots, s_m\}$ be a set of *m* user sessions discovered by preprocessing the web log data, where each user session $s_i \in S$ can be represented as $s = \{w_{u_1}, w_{u_2}, \ldots, w_{u_m}\}$. Each w_{u_i} may be either a binary or non-binary value depending on whether it represents presence and absence of the URL in the session or some other feature of the URL. If w_{u_i} represents presence of absence of the URL in the session, then each user session is represented as a bit vector where

$$w_{u_i} = \begin{cases} 1; \text{ if } u_i \in s; \\ 0; \text{ otherwise} \end{cases} \quad (1)$$

Instead of binary weights, feature weights can also be used to represent a user session. These feature weights may be based on frequency of occurrence of a URL reference within the user session, the time a user spends on a particular page or the number of bytes downloaded by the user from a page.

A. k-Means Clustering Algorithm:

The *k*-Means clustering or Hard *c*-Means clustering algorithm [15] is one of the most commonly used methods for partitioning the data. Given a set of *m* data points $X = \{x_i \mid i = 1 \cdots m\}$, where each data point is a *n*-dimensional vector, *k*-means clustering algorithm aims to partition the *m* data points into *k* clusters ($k \le m$) $\boldsymbol{C} = \{c_1, c_2, \ldots, c_k\}$ so as to minimize an *objective function* (or a cost function) $J(V, X)$ of dissimilarity [29], which is the within-cluster sum of squares. In most cases the dissimilarity measure is chosen as the Euclidean distance. The objective function is an indicator of the distance of the *n* data points from their respective cluster centers.

In most cases this dissimilarity measure is chosen as the Euclidean distance. The objective function *J*, based on the Euclidean distance between a data point vector x_i in cluster *j* and the corresponding cluster center v_j, can be defined by:

$$J(X,V) = \sum_{j=1}^{k} J_i(x_i, v_j) = \sum_{j=1}^{k} \left(\sum_{i=1}^{m} u_{ij} . d^2(x_i, v_j) \right), \quad (2)$$

where, $J_i(x_i, v_j) = \sum_{i=1}^{m} u_{ij} . d^2(x_i, v_j)$,

is the objective function within cluster c_i,

$u_{ij} = 1$, if $x_i \in c_j$ and 0 otherwise.

$d^2(x_i, v_j)$ is the disatnce between x_i and v_j

The Euclidian distance measure to calculate the distance between various data points and cluster centers is given by:

$$d^2(x_i, v_j) = \left\| \sum_{k=1}^{n} x_k^i - v_k^j \right\|^2 \quad (3)$$

where , n is the number of dimensions of each data point

x_k^i is the value of k^{th} dimensions of x_i

v_k^j is the value of k^{th} dimensions of v_j

The k-means clustering first initializes the cluster centers randomly. Then each data point x_i is assigned to some cluster v_j which has the minimum distance with this data point. Once all the data points have been assigned to clusters, cluster centers are updated by taking the weighted average of all data points in that cluster. This recalculation of cluster centers results in better cluster center set. The process is continued until there is no change in cluster centers.

The partitioned clusters are defined by a $m \times k$ binary membership matrix $\boldsymbol{U}$, where the element u_{ij} is 1, if the ith data point x_i belongs to the cluster j, and 0 otherwise. Once the cluster centers $V = \{v_1, v_2, \ldots, v_k\}$, are fixed, the membership function u_{ij} that minimizes (2) can be derived as follows:

$$u_{ij} = \begin{cases} 1; \text{ if } d^2(x_i, v_j) \le d^2(x_i, v_{j^*}) \; j \ne j^*, \forall \; j^* = 1, \cdots, k \\ 0; \text{ otherwise} \end{cases} \quad (4)$$

The equation (4) specifies that assign each data point x_i to the cluster c_j with the closest cluster center v_j. Once the membership matrix $U=[u_{ij}]$ is fixed, the optimal center v_j that minimizes (2) is the mean of all the data point vectors in cluster j :

$$v_j = \frac{1}{|c_j|} \sum_{i, x_i \in c_j}^{m} x_i \quad (5)$$

where ,

$|c_j|$, is the size of cluster c_j and also $|c_j| = \sum_{i=1}^{m} u_{ij}$

The input to the algorithm is a set of m data points $X = \{x_i \mid i = 1 \cdots m\}$, where each data point is a n-dimensional vector, it then determines the cluster centers v_j and the membership matrix $\boldsymbol{U}$ iteratively using the following algorithm as described in Fig. 1.

Algorithm: *k*-Means Clustering
Input: No. of clusters k and Set of *m* data points $X=\{x_1, \ldots, x_m\}$
Output: Set of *k* centroids, $V=\{v_1, \ldots, v_k\}$, corresponding to the clusters $C=\{c_1, \ldots, c_k\}$, and membership matrix $U=[u_{ij}]$.
Steps:
1) Initialize the *k* centroids $V=\{v_1, \ldots, v_k\}$, by randomly selecting *k* data points from *X*.
2) **repeat**
   i) Determine the membership matrix *U* using (4), by assigning each data point x_i to the closest cluster c_j.
   ii) Compute the objective function $J(X,V)$ using (2). Stop if it below a certain threshold ε.
   iii) Recompute the centroid of each cluster using (5).
3) **until** Centroids do not change

Figure 1. *k*-Means Clustering Algorithm

The *k*-means algorithm provides locally optimal solutions with respect to the sum of squared errors represented by the error objective function. Since it is a fast iterative algorithm, it has been applied to a variety of areas [30]-[32].

The attractiveness of the k-means lies in its simplicity and flexibility. However, it suffers from major shortcomings that have been a cause for it not being implemented on large datasets. The most important among these are i) *k*-Means algorithm scales poorly with respect to the time it takes for large number of points; ii) The algorithm might converge to a solution that is a local minimum of the objective function and iii) The main disadvantage of this algorithm lies in its sensitivity to initial positions of the cluster centroids [33]. Since the performance of the k-Means algorithm depends on the initial positions of the cluster centeroids, it is recommended to execute the algorithm multiple times, each with a different set of initial centroids.

B. Global k-Means Clustering Algorithm:

Algorithm: Global *k*-Means Clustering
Input: No. of clusters k and Set of *m* data points $X=\{x_1, \ldots, x_m\}$
Output: Set of *k* centroids, $V=\{v_1, \ldots, v_k\}$, corresponding to the clusters $C=\{c_1, \ldots, c_k\}$.
for j := 1 to *k*
begin
  if j = 1 **then**
    v_1 (1) := centroid of dataset *X*
    V_1:= $\{v_1(1)\}$
  else
    for i := 1 to *m*
    begin
      Execute k-Means using $\{v_1(j\text{-}1), \ldots, v_{j-1}(j\text{-}1), \; x_i \}$ as initial centers
      $v_j^i := j^{th}$ cluster center obtained using (5)
      J_i := objective function value using (2)
    end
  endif
  v_j(j) := v_j^k, where, $1 \le k \le m$ and $J_k = \min_{i=1}^{m} J_i$
  $V_j := \{v_1(j\text{-}1), \ldots, v_{j-1}(j\text{-}1), \; v_j(j)\}$
end

Figure 2. Global *k*-Means Clustering Algorithm

Global k-means clustering algorithm provides a deterministic global optimization that does not depend on the initial positions of cluster centers. It makes use of the *k*-Means algorithm as a local search procedure. Instead of randomly selecting initial values for all cluster centers, the algorithm proceeds in an incremental way and at each step optimally adds one new cluster center.

In order to solve a clustering problem with c clusters the algorithm proceeds as follows:

Starting with one cluster (k = 1), the optimal position of the cluster center is chosen at the centroid of the data set $X=\{x_1, \ldots, x_m\}$. To solve the problem with two clusters (k =2) the first initial cluster center is always placed at the optimal position obtained by solving the problem with k = 1. The *k*-Means algorithm is executed m times. During each iteration i, the data point x_i is chosen as the initial second cluster center. The best solution obtained after the m executions of the k-means algorithm is considered as the solution for the clustering problem with k = 2. Let $\{v_1(k), \ldots, v_k(k)\}$ represents the final solution for k-clustering problem. Proceeding in the same way, Once the $(k-1)$ clustering problem is solved, the solution of the k-clustering problem is

obtained as follows: Execute the k-Means algorithm with k clusters m times, where each run i starts with the initial cluster centers $\{v_1(k\text{-}1), \ldots, v_{k-1}(k\text{-}1), x_i\}$. The best solution obtained from the m runs is considered as the solution $\{v_1(k), \ldots, v_k(k)\}$ of the k-clustering problem. Proceeding in the same way the solution with c-clusters can be obtained.

An advantage of Global k-Means clustering method is that it directly provides clustering solutions for all intermediate values of $k < c$, without requiring any additional computational effort.

C. Fast Global k-Means Clustering Algorithm:

Global k-Means clustering algorithm requires m executions of the k-Means algorithm for each data point x_i in X. Therefore computational time complexity of Global k-means algorithm is rather higher.

The fast global k-Means algorithm accelerates the global k-Means algorithm [4]. Given the solution of the $(k-1)$-clustering problem $\{v_1(k\text{-}1), \ldots, v_{k-1}(k\text{-}1)\}$ and corresponding value of the objective function $J(k\text{-}1)$ representing the sum of the squared error (SSE) as described in (2), the algorithm does not execute the k-Means algorithm for each data point repeatedly to find the solution of the k –clustering problem. Instead, the algorithm computes the upper bound $J(k) \le J(k-1) - b_i$ on the resulting error $J(k)$ for each possible data point $x_i \in X$, where $J(k\text{-}1)$ is the error value of $(k\text{-}1)$-clustering problem and b_i is defined as:

$$b_i = \sum_{j=1}^{m} \max\left(d_{k-1}^{j} - \left\| x_i - x_j \right\|^2, 0 \right), \forall\ i = 1 \ldots m \qquad (6)$$

Here ,

$$d_{k-1}^{j} = \min\left(\left\| x_j - v_1^{k-1} \right\|^2, \ldots, \left\| x_j - v_{k-1}^{k-1} \right\|^2 \right)$$

d_{k-1}^{j} is the closest distance between the squared distance between x_j and the closet center among $(k\text{-}1)$-cluster centers $\{v_1(k\text{-}1), \ldots, v_{k-1}(k\text{-}1)\}$, that is, the squared distance between x_j and the center of a cluster it belongs to. m is the size of the data set. A data point $x_i \in X$ with the maximum value of b_i is chosen as an initial center for the k^{th} cluster center.The quantity b_i measures the guaranteed reduction in the error measure obtained by inserting a new cluster center at position x_i.

Algorithm: Fast Global k-Means Clustering
Input: Set of m data points $X=\{x_1, \ldots, x_m\}$.
Output: Set of k centroids, $V=\{v_1, \ldots, v_k\}$, corresponding to the clusters $C=\{c_1, \ldots, c_k\}$.

```
for j := 1 to k
begin
    if j = 1 then
    v1 (1) := centroid of dataset X
        V1:= {v1(1)}
    else
        for i := 1 to m
        begin
            Compute bi using (6)
        end
    endif
    k := arg max_i bi
    Using {v1(k-1), ..., vk-1(k-1), xk} as initial cluster centers,
    execute the k-Means algorithms to obtain the solution Vj := {v1(j-
    1), ..., vj-1(j-1), vj(j)}
end
```

Figure 3. Fast Global k-Means Clustering Algorithm

IV. Experimental Results

In order to discover the clusters that exist in user accesses sessions of a web site, we carried out a number of experiments. The Web access logs were taken from the P.A. College of Engineering, Mangalore web site, at URL http://www.pace.edu.in. The site hosts a variety of information, including departments, faculty members, research areas, and course information. The Web access logs covered a period of one month, from February 1, 2011 to February 8, 2011. There were 12744 logged requests in total.

After performing the cleaning step the output file contains 12744 entries. Total numbers of unique users identified are 16 and the number of user sessions discovered are 206. Table II depicts the results of cleaning and user identification steps.

TABLE II

RESULTS OF CLEANING AND USER IDENTIFICATION

Items	Count
Initial No of Log Entries	12744
Log Entries after Cleaning	11995
No. of site ULRs accessed	116
No of Users Identified	16
No. of User Sessions Identified	206

Figure 4 shows the result of user session identification. It depicts the percentage of user sessions accessing the specified number of URLs.

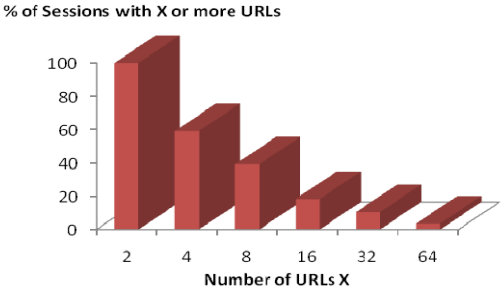

Figure 4. Percentage of Sessions accessing X No. of URLs

Once the user sessions are discovered, k-Means, Global k-Means and Fast Global k-Means clustering algorithms are applied to discover session clusters that represent similar URL access patterns. In order to discover the user session clusters, we conducted the following experiments:

1. Multiple runs of k-Means algorithm with number of clusters ranging from $k = 2, \ldots, 67$. (The value 67 for the number of clusters is one third of total number of the discovered user sessions).
2. One run of the Global k-Means algorithm for $k = 67$.
3. One run of the Fast Global k-Means algorithm for $k = 67$.

For each of the above runs we computed the value of the objective function (J) using (2), which represents the sum of the squared error. We also computed the execution timings for all of the above runs.

Since the above clustering algorithms result in different clusters it is important to perform an evaluation of the results to assess their quality. We evaluated our results based on DB index [34], which is a quality measure that tries to minimize the intra-cluster scatter while maximizing the inter-cluster separation in order to find compact and well separated clusters. This validity measure is described below:

Davies-Bouldin Validity Index: Davies and Bouldin attempts to minimize the average distance between each cluster and the one most similar to it. It is defined as:

$$DB = \frac{1}{k}\sum_{i=1}^{k} \max_{1\le j\le k, j\neq i}\left(\frac{diam(c_i)+diam(c_j)}{dis(c_i,c_j)}\right) \qquad (8)$$

An optimal value of the k is the one that minimizes this index.

Clusters formed by the applications clustering algorithms represent a group of user sessions that are similar based on co-occurrence patterns of URL references. Clustering of user sessions results in a set $C=\{c_1,c_2,\ldots,c_k\}$ of clusters, where each c_i is a subset of S, i.e., a set of user sessions. Each cluster represents a group of users with similar navigational patterns.

Table III describes the clustering results after the application of all the three kinds of clustering algorithms.

TABLE III
CLUSTERING RESULTS

Evaluation Measure	Clusters	k-Means	Global k-Means	Fast Global k-Means
Sum of Squared Error (*J*)	10	583.54	504.80	521.56
	20	443.06	358.95	376.22
	30	357.24	275.72	295.47
	40	284.09	208.01	228.83
	50	279.29	162.52	176.12
	60	260.64	127.17	139.36
DB Index	10	1.3395	1.2827	1.070
	20	1.3456	1.1295	0.9526
	30	1.2228	0.8982	0.6880
	40	1.1045	0.8038	0.6683
	50	1.1345	0.7374	0.6839
	60	0.8846	0.6401	0.5951
Execution Time (milli seconds)	10	49	38369	2071
	20	110	113623	4481
	30	142	213973	7004
	40	164	335883	9728
	50	278	478573	12662
	60	188	644976	15807

The graph plot in Fig. 5 displays the clustering error as a function of the number of clusters. It is clear that the global k-means algorithm provides the solutions of equal or better quality with respect to the k-means algorithm. Fast Global *k*-Means Algorithm also provides solutions of excellent quality, comparable to those obtained by the Global k-Means method.

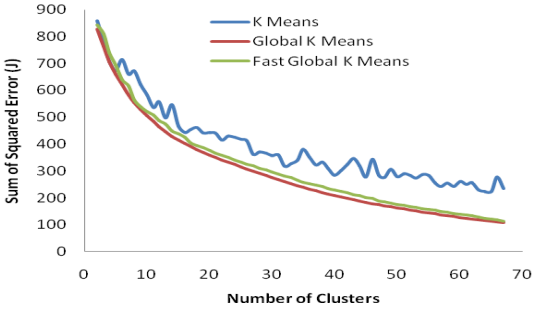

Figure 5. Clustering Error (*J*) versus No. of Clusters

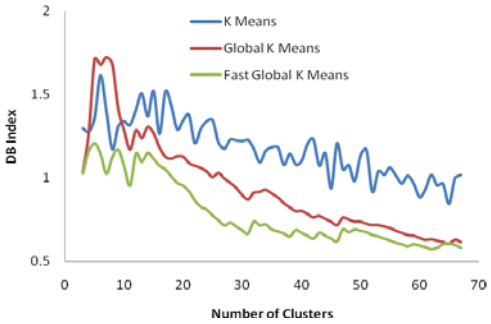

Figure 6. DB Validity Index versus No. of Clusters

The graph plot in Fig. 6 displays the DB validity index value as a function of the number of clusters. It shows that the solution provided by Fast Global k-means algorithm outperforms the other two algorithms. The Global *k*-Means Algorithm also provides solutions of excellent quality, much better than those obtained by the k-Means method. The graph plot in Fig. 7 provides the execution time in second as a function of the number of clusters. It is clear that the Fast Global k-means algorithm provides the solutions much faster than the Global k-means algorithm. Fast Global *k*-Means Algorithm also provides solutions of excellent quality, comparable to those obtained by the Global k-Means method.

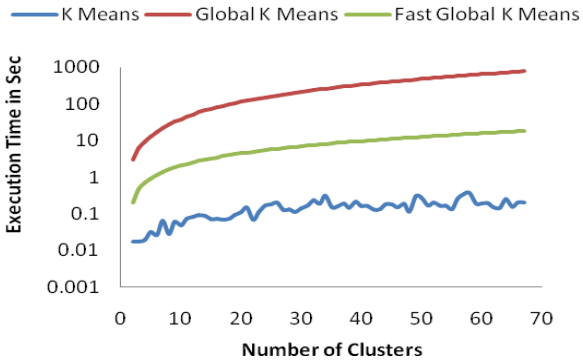

Figure 7. Execution Time versus No. of Clusters

V. CONCLUSION AND FUTURE WORK

In this paper we have presented our framework for web usage data clustering for users' sessions using variants of k-Means clustering algorithms. We provided a detailed overview of the techniques to pre-process the web log data including data cleaning, user identification and session identification. We also described the mathematical model and algorithm details about how to apply the k- Means, Global *k*-Means, and Fast Global k-Means Clustering algorithm in order to cluster the user sessions.

Global *k*-means clustering algorithm is a deterministic clustering method, independent of any starting conditions and provides excellent results in terms of the sum of the squared error criterion. The method performs much better than the *k*-Means algorithm with multiple random restarts. Another advantage of the Global *k*-means clustering technique is that in order to solve the *c*-clustering problem, all intermediate *k*-clustering problems are also solved for $k = 1, \ldots, c$. This is especially useful in applications where we seek for the actual

number of clusters and for that *k*-clustering problem is solved for several values of k.

The Fast Global *k*-Means algorithm significantly reduces the required computational effort, while at the same time providing solutions of almost the same clustering error quality. Thus Fast Global *k*-Means algorithm is much more scalable than Global *k*-Means algorithm.

Another direction of future work is related with the use of fuzzy *c*-Mean clustering technique to discover the user session clusters. The reason behind this is, although different variants of *k*-Means clustering algorithm are efficient in handling the crisp data which have clear cut boundaries, but in reality web usage data is semi-structured and contains the outliers and incomplete navigational data, due to a wide variety of reasons inherent to web browsing and logging. Therefore, Web Usage Mining requires modelling of multiple overlapping sets in the presence of significant noise and outliers. Soft Computing based techniques such as Fuzzy Clustering can be very useful for mining such semi structured, noisy and incomplete data.

REFERENCES

[1] B. Mobasher. Data mining for web personalization. Lecture Notes in Computer Science, 4321:90, 2007.

[2] R. Cooley, B. Mobasher, and J. Srivastava, "Web mining: Information and pattern discovery on the World Wide Web" in Ninth IEEE International Conference on Tools with Artificial Intelligence, 1997. Proceedings. 1997, pp. 558–567.

[3] Y. Fu, K. Sandhu, and M. Shih, "A generalization-based approach to clustering of web usage sessions," Lecture Notes in Computer Science, pp. 21–38, 2000.

[4] A. Likas, M. Vlassis, and J. Verbeek, "The global k-means clustering algorithm," Pattern Recognition, vol. 36, pp. 451–461, 2003.

[5] M. K. Jiawei Han, Data Mining: Concepts and Techniques. Academic Press, Morgan Kaufmarm Publishers, 2001.

[6] P. S. U. M. Fayyad, G. Piatetsky-Shapiro and E. R. Uthurusamy, "Advances in knowledge discovery and data mining," in CA: AAAI/MIT Press, 1996.

[7] J. H. Ming-Syan Chen, "Data mining an overview from database perspective," Knowledge and data Engineering, IEEE Transactions on, vol. 8, 1996.

[8] J. Srivastava, R. Cooley, M. Deshpande, and P.N. Tan. Web usage mining: Discovery and applications of usage patterns from web data. SIGKDD explorations, 1(2):12–23, 2000.

[9] Etzioni O. Perkowitz, M. Adaptive web sites: Automatically synthesizing web pages. In Proceedings of the 15th National Conference on Artificial Intelligence, Madison, WI (July1998) 727-732, 1998.

[10] Etzioni O. Perkowitz, M. Adaptive web sites. Communications of ACM, 43:152–158, 2000.

[11] Ajith Abraham. Business intelligence from web usage mining. Journal of Information & Knowledge Management, 2(4):375–390, 2003.

[12] Edith Cohen, Balachander Krishnamurthy, and Jennifer Rexford. Improving end-to-end performance of the web using server volumes and proxy filters. SIGCOMM Comput. Commun. Rev., 28:241–253, October 1998.

[13] Alexandros Nanopoulos, Dimitrios Katsaros, and Yannis Manolopoulos. Exploiting web log mining for web cache enhancement. In WEBKDD 2001 Mining Web Log Data Across All Customers Touch Points, volume 2356 of Lecture Notes in Computer Science, pages 235–241. Springer Berlin / Heidelberg, 2002.

[14] G. Vigna, W. Robertson, Vishal Kher, and R.A. Kemmerer. A stateful intrusion detection system for world-wide web servers. In Computer Security Applications Conference, 2003. Proceedings. 19th Annual, pages 34–43, 2003.

[15] J. A. Hartigan and M. A. Wong, "A k-means clustering algorithm, Applied Statistics, 28:100--108, 1979.

[16] D. Tanasa and B. Trousse, "Data preprocessing for WUM" Potentials, IEEE, vol. 23, no. 3, pp. 22 – 25, 2004.

[17] R. W. Cooley, "Web usage mining: Discovery and application of intresting patterns from web data," Ph.D. dissertation, Th e Graduate School of the University of Minnesota, 2000.

[18] D. Tanasa and B. Trousse, "Advanced data preprocessing for intersites web usage mining," IEEE Intelligent Systems, vol. 19, no. 2, pp. 59–65, 2004.

[19] R. Cooley, B. Mobasher, J. Srivastava et al., "Data preparation for mining world wide web browsing patterns," Knowledge and Information Systems, vol. 1, no. 1, pp. 5–32, 1999.

[20] P.-N. Tan and V. Kumar, "Discovery of web robot sessions based on their navigational patterns," Data Min. Knowl. Discov., vol. 6, pp. 9–35, January 2002

[21] P. Pirolli, J. Pitkow, and R. Rao, "Silk from a sow's ear: extracting usable structures from the web," in Proceedings of the SIGCHI conference on Human factors in computing systems: common ground, ser. CHI '96. New York, NY, USA: ACM, 1996, pp. 118–125.

[22] M. Spiliopoulou, B. Mobasher, B. Berendt, and M. Nakagawa, "A framework for the evaluation of session reconstruction heuristics in webusage analysis," INFORMS J. on Computing, vol. 15, pp. 171–190, April 2003.

[23] Bettina Berendt, Bamshad Mobasher, Miki Nakagawa, and Myra Spiliopoulou. The impact of site structure and user environment on session reconstruction in web usage analysis. In WEBKDD 2002 - MiningWeb Data for Discovering Usage Patterns and Profiles, volume 2703 of Lecture Notes in Computer Science, pages 159–179. Springer Berlin / Heidelberg, 2003.

[24] Bettina Berendt and Myras Spiliopoulou. Analysis of navigation behaviour in web sites integrating multiple information systems. The VLDB Journal, 9:56–75, 2000.

[25] Lara D. Catledge and James E. Pitkow. Characterizing browsing strategies in the world-wide web. Computer Networks and ISDN Systems, Proceedings of the Third International World-Wide Web Conference. 27(6):1065–1073, 1995.

[26] P. Berkhin, "Survey of clustering data mining techniques," Springer, 2002.

[27] B. Pavel, "A survey of clustering data mining techniques," in Grouping Multidimensional Data. Springer Berlin Heidelberg, 2006, pp. 25–71.

[28] R. Xu and I. Wunsch, D., "Survey of clustering algorithms," Neural Networks, IEEE Transactions on, vol. 16, no. 3, pp. 645–678, May 2005.

[29] Jang, J.-S. R., Sun, C.-T., Mizutani, E., "Neuro- Fuzzy and Soft Computing – A Computational Approach to Learning and Machine Intelligence," Prentice Hall.

[30] R.O. Duda, P.E. Hart, Pattern Classification and Scene Analysis, Wiley, New York, 1973.

[31] A.K. Jain, R.C. Bubes, Algorithm for Clustering Data, Prentice-Hall, Englewood Cli s, NJ, 1988.

[32] L. Kaufman, P.J. Rousseeuw, Finding Groups in Data. An Introduction to Cluster Analysis, Wiley, New York, 1990.

[33] P.S. Bradley and Usama M. Fayyad: Refining initial points for k-means clustering. In Proceedings Fifteenth International Conference on Machine Learning, pages 91-99, San Francisco, CA, 1998, Morgan Kaufmann.

[34] D.L. Davies, D.W. Bouldin. A cluster separation measure. 1979. IEEE Trans. Pattern Anal. Machine Intell. 1 (4). 224-227.