# Research on Web Session Clustering

Li Chaofeng
College of Management, South-Central University for Nationalities, Wuhan , P.R. China
Email: licfonly@126.com

***Abstract*—The task of clustering web sessions is to group web sessions based on similarity and consists of maximizing the intra-group similarity while minimizing the inter-group similarity. The results of Web session clustering can be used in personalization, system improvement, site modification, business intelligence, usage characterization and so forth. This paper proposes a framework of Web session clustering first. Then several data preparation techniques that can be used to improve the performance of data preprocessing are presented. A new method for measuring similarities between web pages that takes into account not only the URL but also the viewing time of the visited web page is also introduced and a new method to measure the similarity of web sessions using sequence alignment and the similarity of web page access is given in detail. Finally, an algorithm of web session clustering is proposed. This algorithm defines the number of clusters according to the knowledge of application fields, takes advantage of ROCK to decide the initial data points of each cluster and determines the criterion function according to the contributions of overall increase in similarities made by dividing Web sessions into different clusters --- which not only overcomes the shortcomings of traditional clustering algorithm which merely focus on partial similarities, but also decreases the complexities of time and space.**

***Index Terms*—Web session clustering; Data Preprocessing; sequence alignment; similarity measurement***

## I. INTRODUCTION

Clustering is a useful technique for grouping data points such that points within a single cluster have similar characteristics while points in different groups are dissimilar. From a practical perspective clustering plays an outstanding role in data mining applications such as scientific data exploration, information retrieval and text mining, spatial database applications, Web analysis, CRM, marketing, medical diagnostics, computational biology, and many others[1]. Since it's the core of pattern discovery algorithm especially cluster, clustering web sessions is very important for Web usage mining which is the application of data mining techniques to discover usage patterns from Web data, in order to understand and better serve the needs of Web-based applications[2].

Generally, Web session clustering consists of three processes: data preprocessing, measurement of similarity between web sessions and algorithm of Web session clustering.

Data preprocessing is the process to convert the raw data into the data abstraction necessary for the further applying the data mining algorithm. As the data sources of Web session clustering, the results' quality of data preprocessing influences the results of Web session clustering directly. So, it is particularly important for Web session clustering processes[3].

The first and foremost question needed to be considered in clustering web sessions is how to measure the similarity between web sessions. Generally, the more the precision of similarity measurement is, the better the results of web session clusters are and vice versa. Most of the previous related works apply either Euclidean distance for vector or set similarity measures, Cosine or Jaccard Coefficient. There are many shortcomings for them. First, web sessions must be transferred in order to use them. However, the transferred space could be in very high dimension. Secondly, the original click stream cannot be fully represented by a vector or a set of URLs where the order of clicks is not considered. Finally, most of the data in Web session clustering are categorical. However, Euclidean distance has been proven in practice not suitable for measuring similarity in categorical vector space[1].

When we take into account a clustering algorithm, there are 3 key points and difficulties that we have to chew over: (1) How many clusters are desired? (2) Are the initial data points in cluster suitable? The initial points that divided into the cluster must be exact, or the quality of the clusters will be not so good. (3) How to merge the rest data points in the data set into different clusters? This is the core question of a clustering algorithm, which determines the quality of the clusters.

## II. FRAMEWORK OF WEB SESSION CLUSTERING

### A. Data Source of Web Session Clustering

Web session clustering's data source can be collected at the server-side, client-side, proxy servers, or obtained from an organization's database which contains business data or consolidated Web data.

- Server Level Collection

The data recorded in server logs are very important for performing Web session clustering because they reflect the access of a Web site by multiple users explicitly. However, the site usage data recorded by server logs may

Foundation item: Supported by Natural Science Foundation of South-Central University for Nationalities(YZY06009)

not be entirely reliable due to many reasons, such as the cached page views, use of POST method to transfer information and use of cookies having raised growing concerns regarding user privacy etc. Otherwise, the Web server also relies on other utilities such as CGI scripts to handle data sent back from client browsers.

- Client Level Collection

Client-side data collection can be implemented by two ways: using a remote agent such as Java scripts or Java applets and modifying the source code of an existing browser such as Mozilla to enhance its data collection capabilities. Client-side collection has an advantage over server-side collection because it improves both the caching and session identification problems. However, the implementation of which requires user cooperation. Java applets may generate some additional overhead especially when they are loaded for the first time. Java scripts cannot capture all user clicks (such as reload or back buttons) either. Modifying the source code of an existing browser must convince the users to use the modified browser for their daily browsing activities, but it is very difficult.

- Proxy Level Collection

A Web proxy acts as an intermediate level of caching between client browsers and Web servers. Proxy caching can be used to reduce the loading time of a Web page experienced by users as well as the network traffic load at the server and client sides [4]. Proxy traces may reveal the actual HTTP requests from multiple clients to multiple Web servers. This may serve as a data source for characterizing the browsing behavior of a group of anonymous users sharing a common proxy server.

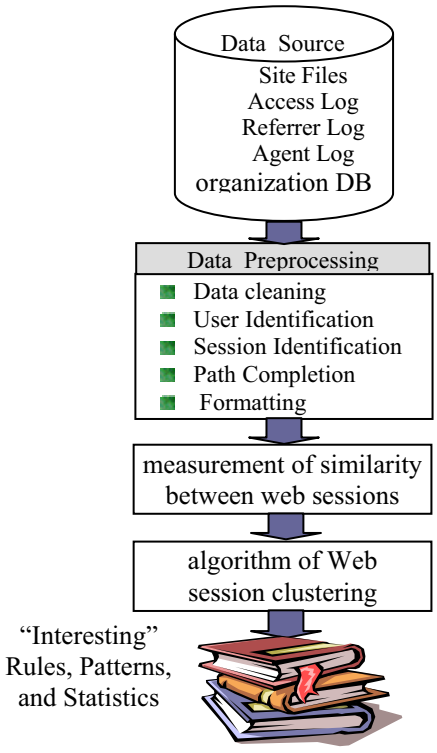


Figure 1. Framework of Web Session Clustering

### B. *Framework of Web Session Clustering*

The primary objective of Web session clustering is to discover interesting patterns in accesses to various Web pages within the Web space associated with a particular server. Generally, it comprises three phases, namely preprocessing, measurement of similarity between web sessions and algorithm of Web session clustering, as shown in Fig. 1.

## III. DATA PREPROCESSING IN WEB SESSION CLUSTERING

Ideally, the input for the measurement algorithm of similarity between web sessions is a user session file that gives an exact accounting of who accessed the Web site, what pages were requested and in what order, and how long each page was viewed. However, because of the reasons we will discuss in the following, the information contained in a raw Web server log does not reliably represent a user session file before data preprocessing. Generally, data preprocessing consists of data cleaning, user identification, session identification and path completion, as shown in Fig. 2.

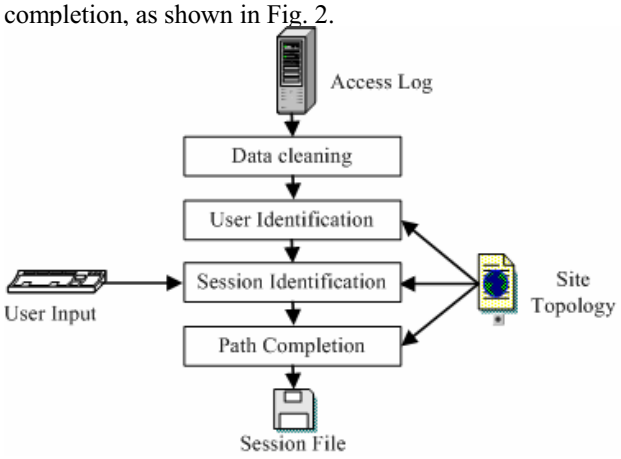


Figure 2. Phases of Data Preprocessing in Web session clustering

### A. *Data Cleaning*

The task of data cleaning is to remove the irrelevant and redundant log entries for the mining process. There are three kinds of irrelevant or redundant data need to clean. First, accessorial resources embedded in HTML file should be removed. HTTP protocol is connectionless. 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. Since the main intent of Web session clustering is to get a picture of the user's behavior, it does not make sense to include file requests that the user did not explicitly request. Elimination of the items deemed irrelevant can be accomplished by checking the suffix of the URL name. The second is robots' requests. Web robots (also called spiders) are software tools that scan a Web site to extract its content. spiders automatically follow all the hyperlinks from a Web page. Search engines such as Google periodically use spiders to grab all the pages from a Web site to update their search indexes[5]. To remove robots' request, we can Look for all hosts that have requested the page "robots.txt". Finally, Error's requests are useless for mining process. They can

be removed by checking the status of request. Data cleaning can be accomplished by Fig. 3.

```
Datacleaning (LogFile: Web log file; LogBase: Web log database) {
 while not eof (LogFile ) {
  LogRecord=Read (LogFile)
   if ((LogRecord.Cs-url-stem!=(gif,jpegjpg,cssjs))&&
        (LogRecord.Cs-
        mehod=="GET")&&(LogRecord.Sc-
        status!=(301,404,500)&&(LogRecord.User-agent !=
        Crawler,Spide,Robot))
        Write (LogBase,LogRecord )
  }
}
```

Figure 3. Algorithm of data cleaning

### B. User Identification

A user is defined as the principal using a client to interactively retrieve and render resources or resource manifestations. User identification is greatly complicated by the existence of local caches, corporate firewalls, and proxy servers. The Web session clustering methods that rely on user cooperation are the easiest ways to deal with this problem. However, it's difficulty because of security and privacy. In our experiment, we use the following heuristics to identify the user: 1) Each IP address represents one user; 2) For more logs, if the IP address is the same, but the agent log shows a change in browser software or operating system, an IP address represents a different user; 3) Using the access log in conjunction with the referrer logs and site topology to construct browsing paths for each user. If a page is requested that is not directly reachable by a hyperlink from any of the pages visited by the user, there is another user with the same IP address. Thus, the process of user identification can be described by Fig. 4.

```
UserIden(LogBase: Web log database; UserBase: User datatbse) {
   IPSet=φ
   Userset=φ
   Browserset=φ,
   OSSet=φ
   i=0
   while not eof ( LogBase ) {
      LogRecord=Read (LogBase )
      if LogRecordJP not in IPSet {
         IPSet=IPSet ∪(LogRecord.IP)
         BrowserSet=BrowserSet∪{LogRecord.Browser}
         OSSet=OSSet∪{ LogRecord.OS }
         i=i+1
         UserSet=UserSet ∪{Ui}
      }else
         if LogRecord.Browser not in Browserset ||
            LogRecord.OS not in OSSet {
           i=i+1
           UserSet=UserSet U {Ui}
         }
      }
   }
}
```

Figure 4. Algorithm of user identification

### C. User Session Identification

A user session means a delimited set of user clicks (click stream) across one or more Web servers. The goal of session identification is to divide the page accesses of each user into individual sessions. At present, the methods to identify user session include timeout mechanism[6] and maximal forward reference[7] mainly. The following is the rules we use to identify user session in our experiment:

1) If there is a new user, there is a new session;

2) In one user session, if the refer page is null, there is a new session;

3) If the time between page requests exceeds a certain limit(30 or 25.5mintes), it is assumed that the user is starting a new session.

### D. Path Completion

As the existence of local cache and proxy server, there are many important accesses that are not recorded in the access log. The task of path completion is to fill in these missing page references. Methods similar to those used for user identification can be used for path completion. If a page request is made that is not directly linked to the last page a user requested, the referrer log can be checked to see what page the request came from. If the page is in the user's recent request history, the assumption is that the user backtracked with the "back" button available on most browsers, calling up cached versions of the pages until a new page was requested. If the referrer log is not clear, the site topology can be used to the same effect. If more than one page in the user's history contains a link to the requested page, it is assumed that the page closest to the previously requested page is the source of the new request. Fig. 5 describes the process of session identification with path completion.

```
SessionIden(LogBase: Web log database; SessionBase: Session database ) {
   Sessionset=φ
   UserSet=φ
   k=0
   while not eof (LogBase ) {
      LogRecord=Read (LogBase)
      if (LogRecord.Refer= '-' || LogRecord.time-taken>30min || LogRecord.UserID not in UserSet) {
            k=k+ 1
            Sk=LogRecord.Ur1
            Sessionset= Sessionset U {Sk}
         }else
            for i=l to k {
               Di=Distance (LogRecord.Url,Si)
               if Di=is the MINIMUM {
                     Si=Si + LogRecord.Ur1
                     CompletePath(Si, TopoBse)
               }
            }
      }
   }
}
```

Figure 5. Algorithm of path completion

## IV. MEASUREMENT OF SIMILARITY BETWEEN WEB SESSIONS

The first and foremost question needed to be considered in clustering web sessions is how to measure the similarity between two web sessions. Most of the previous related works apply either Euclidean distance for vector or set similarity measures, Cosine or Jaccard Coefficient. For example, Shahabi et al introduced a novel path clustering method based on the similarity of the history of user navigation. Similarity between each two paths is measured by the definition of Path Angle which is actually based on the Cosine similarity between two vectors[8]. Fu et al cluster users based on clustering web sessions[9]. Their method scaled well over increasing large data. Mobasher et al. used the Cosine coefficient and a threshold of 0.5 to cluster on a web log [10]. Banerjee and Ghosh introduced a new method for measuring similarity between web sessions: The longest common sub-sequences between two sessions is first found through dynamic programming, then the similarity between two sessions is defined through their relative time spent on the longest common sub-sequences[11]. Wang et al. consider each session as a sequence and borrow the idea of sequence alignment in bioinformatics to measure similarity between sequences of page accesses[1]. Sequence alignment is one of fundamental operations in bioinformatics. Many researchers have focused on this field[12] since Needleman and Wunsch had presented this technique applying to the search for similarities in the amino acid sequences of two proteins in 1970[13].

In this paper, we propose a method for measuring similarities between web pages that takes into account not only URL but also viewing time of the visited web page.

- Similarity Between Web Pages Based on Pages' URL

The details of this method were discussed in [1]. The content of pages is not considered but simply the paths leading to a web page (or script). Here is an example to illustrate the idea. Suppose "/Lab/sjsx/shsjsx/bysx.htm" and "/Lab/sjsx/sxgztl.htm" are two requested pages in web log, Each level of a URL can be represented by a token. Thus, the token string of the full path of a URL is the concatenation of all the representative tokens of each level. We can get different token strings of the URL correspond with the different requested pages via making a tree structure of the web site. Assuming that the two token strings are "0222" and "021", we compare each corresponding token of the two token strings one by one from the beginning until the first pair of tokens is different. For given example, the process is shown in Fig. 6.

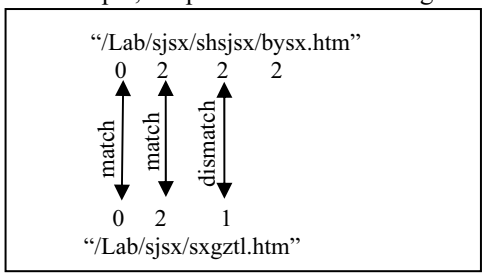


Figure 6. Compare Token Strings

Marking $L_{longer} = (l_1 > l_2)\ ?\ l_1 : l_2$, here $l_1$ and $l_2$ are lengths of the two token strings, then we give weight to each level of the longer token: the last level is given weight 1, the second to the last level is given weight 2, the third to the last level is given weight 3, and so on and so forth, until the first level which is given weight $L_{longer}$. The similarity between two token strings is defined as the sum of the weight of those matching tokens divided by the sum of the total weights. For our example, $L_{longer} = 4$, weight for each level is shown in Fig. 7. The similarity of the two requested web pages is $S_{URL} = (4 + 3)/(4+3+2+1) = 0.7$.

| | | | | |
|---|---|---|---|---|
| Token string 1: | 0 | 2 | 2 | 2 |
| Token string 2: | 0 | 2 | 1 | |
| Weight of each token: | 4 | 3 | 2 | 1 |

Figure 7. Weight Each Token Level

The character of this similarity measurement includes:

1) $0 \le S_{URL} \le 1$, i.e. the similarity of any pair of web pages is between 0.0 and 1.0;

2) $S_{URL} = 0$, when the two web pages are totally different;

3) $S_{URL} = 1$, when the two web pages are exactly same.

- Similarity between Web Page Accesses Based on Viewing Time

*1) Viewing Time*

Interactions between client and server are shown in Fig. 8. Time that a user views a page is from $t_3$ to $t_4$, i.e. $t_{view}=t_4-t_3$.

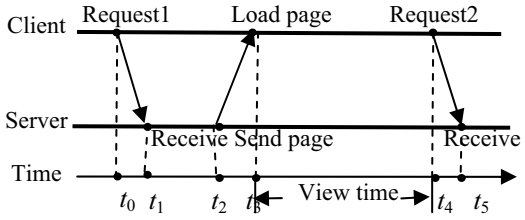


Figure 8. Process of Client/Server Interactions

However, the viewing time recorded in Web sever log is $t_{log}=t_5-t_1$. So, we have to calculate $t_{view}$. This can be done as follows:

First, the duration from Web server received a request to it sent a page to client is very short and usually a fraction of millisecond, so it can be neglected. Thus:

$$t_1 \approx t_2 \tag{1}$$

Second, suppose the round-trip time that Web server executes a ping command to the client is δ, then the time that client sent a request to server can be approximated as δ/2, i.e.

$$t_1 - t_0 = t_5 - t_4 = \delta/2 \tag{2}$$

Last, the time that server sent a page to client can be calculated as (3):

$$t_3 - t_2 = P_S \times \delta/2 \tag{3}$$

Where $P_S$ is the size of the page in number of TCP/IP packets.

Thus, the accurate viewing time spent on a web page can be computed via (4):

$$\begin{aligned} t_{\text{view}} &= t_4 - t_3 \\ &= (t_5 - t_1) - (t_3 - t_1) - (t_5 - t_4) \\ &\approx (t_5 - t_1) - (t_3 - t_2) - (t_5 - t_4) \\ &= t_{\log} - P_S \times \delta/2 - \delta/2 \\ &= t_{\log} - (P_S + 1) \times \delta/2 \end{aligned} \quad (4)$$

*2) Similarity between Web Page Accesses Based on Viewing Time*

Users browse web pages is a dynamic process. For example, to browse page B, one user has to click page A first although he/she isn't interested to it. Furthermore, the difference in the view time may mean a difference in the users' interests in the corresponding pages. Therefore, we should consider the viewing time when measuring the similarity of web pages.

Assuming the viewing time of page A and B is $t_{\text{view}_A}$ and $t_{\text{view}_B}$, we define the similarity of viewing time between page A and B as (5).

$$S_{\text{time}} = \frac{\min(t_{\text{view}_A}, t_{\text{view}_B})}{\max(t_{\text{view}_A}, t_{\text{view}_B})} \quad (5)$$

Suppose $t_{\text{view}_A} \le t_{\text{view}_B}$, thus we can conclude from (4) and (5):

$$\begin{aligned} S_{\text{time}} &= \frac{t_{\text{view}_A}}{t_{\text{view}_B}} \\ &= \frac{2t_{\log_A} - (P_{S_A} + 1) \times \delta_A}{2t_{\log_B} - (P_{S_B} + 1) \times \delta_B} \end{aligned} \quad (6)$$

For any pair of requested pages, the similarity of viewing time is between 0 and 1.

- Similarity Measurement between Web Page Accesses

As mentioned above, web page access is a dynamic process. Each of the similarity of web pages' URLs or viewing time can't reveal the similarity of web page access truly. Therefore, we propose to measure the similarity of web page access via URL structure of web page in conjunction with web page's viewing time.

Suppose similarity of web pages' URL between Page A and Page B is $S_{URL}$ and the similarity of viewing time is $S_{time}$, we define similarity of web page access between A and B as (7).

$$S = \begin{cases} S_{URL}, & S_{URL} \neq 1 \\ S_{time}, & S_{URL} = 1 \end{cases} \quad (7)$$

*B. Similarity Measurement Between Web Sessions*

Each DNA sequences contain a sequence of amino acids, and this is very similar to each session which consists of a sequence of web pages. Naturally, techniques used in DNA or protein sequences alignment can apply to measure the similarity of web session. Since web pages in a session are usually much less than the elements in DNA or protein, some typical problems in DNA or protein sequence alignment such as the tradeoff between memory efficiency and computational efficiency become not so important in web session sequence alignment, because the memories are enough to contain the whole process. Thus, the problem of computing the similarity between web sessions is converted to find the best matching between two web page access sequences, and can be done by using dynamic programming techniques.

We should have a scoring system to find the optimal matching when we use dynamic programming techniques to compute the similarity between web sessions. For each identical matching, i.e. a pair of pages with similarity 1.0, the similarity score is 20; for each mismatching, i.e. a pair of pages with similarity 0.0 or match a page with a gap, the similarity score is -10; For a pair of pages with similarity $\alpha \in (0,1)$, the score for their matching is between -10 and 20. So, we can use (8) as the similarity scoring function between page $p_i$ and $p_j$

$$\sigma(p_i, p_j) = -10 + 30 \times \alpha, \quad 0 \le \alpha \le 1 \quad (8)$$

The algorithm of similarity measurement between web sessions based on sequence alignment is described as Fig. 9. Two session sequences, S[1..m] and T[1..n], are the input arguments. The outputs are the similarities between sessions and are stored in array P.

```
Input: S[1..m],T[1..n]
Output: P[k],S'[1..k],T'[1..k]
Session_Sim(S[1..m], T[1..n]){
  M[0,0]=0;
  for(i=1;i<=m;i++)
    M[i,0]=M[i-1,0]+ σ(S[i],'-');
  for(j=1;j<=n;j++)
   M[0,j]=M[0,j-1]+ σ('-',T[j]);
  for(i=1;i<=m;i++)
    for(j=1;j<=n;j++)
                  { M[i-1,j]+σ(S[i],'-')
      M[i,j]= max { M[i,j-1]+σ('-',T[j])
                  { M[i-1,j-1]+σ(S[i],T[j])
  for(k=max(i,j);k>0;k--){
    P[k]=M[i,j];
    if(M[i,j]= =M[i-1,j-1]+ σ(S[i],T[j])
      {S'[k]=S[i]; T'[k]=T[j]; i--; j--; }
    if(M[i,j]= = M[i-1,j]+ σ(S[i],'-')
      { S'[k]=S[i]; T'[k]= '-'; i--; }
    if(M[i,j]= =M[i,j-1]+ σ('-',T[j])
      { S'[k]=-'-'; T'[k]=T[j]; j--; }
  }
}
```

Figure 9. Algorithm of similarity measurement between web sessions

The algorithm's complexities of time and space both are O(m×n).

After finding the final score for the optimal session alignment, the final similarity between sessions is computed by considering the final optimal score and the length of the two sessions. In our method, we first get the length of the shorter session --$l_{shorter}$, then the similarity between the two sessions is achieved through dividing the optimal matching score by 20* $l_{shorter}$ because the optimal score can not be more than 20* $l_{shorter}$ in our scoring system.

## V. ALGORITHM OF WEB SESSION CLUSTERING

The topic of Web session clustering has become popular in the field of practical application of clustering techniques recent years. As for the algorithm of clustering, Li et al. studied clustering algorithms based on models [14]. They presented the framework of clustering for objects of model and studied the relations between the number of cluster in clustering analysis, the size of ensemble learning, and performance of ensemble learning. Yu et al. proposed a unifying generative framework for partitional clustering algorithms according to a novel definition of the mean, called a general c-means clustering model (GCM) [15]. Li et al. proposed a novel clustering algorithm based on hierarchical and k-means clustering, which has good computational complexity [16].

### *A. Key points of Clustering*

The number of clusters, the initial data points of the respective clusters, and the defining of criterion function are the 3 key points and difficulties that deserve consideration in Web session clustering.

- Determination of clusters' number

How many clusters should be data set divided into? This question is relative to application field and the user's final aims. For example, for the voting system, we may expect two clusters: all "yes" in one and all "no" in the other. However, for the buyer's log in the E-commerce's server, the number of clusters may be relative to the kinds of the commodities and the types of purchase behaviors that to be analyzed. Therefore, the number of the clusters is determined by analyst who will take into account all aspect's factors.

- Merger methods of remained data points

Suppose that user requires k clusters, $C_1$, $C_2$… $C_k$, and there are initial data points in each cluster. How to merge the rest data points in data set into the k clusters is determined by the criterion function. Different merger methods generate different clustering algorithm. For example, k-means determines a data point to be merged into a cluster by the distance between the data point and the means of all data points in a cluster [17]; k-medoids determines by the distance between the data point and a representative point of the cluster [18]; ROCK introduced a novel concept of links to measure the similarity between a pair of data points and maximized the sum of link for data point pairs belonging to a single cluster as well as minimized the sum of links for data point pairs in different clusters [19]. As for Web session clustering, we propose that one Web session should be merged into a session cluster such that the overall increase of similarities between sessions after the data point being merged into this cluster should be maximum. Intuitively, the cluster of maximum increase is the most similar to current session, so this session should be merged into that cluster. According to this simple fact, we propose the criterion function that merging a session into cluster as (9) in this paper.

$$E_i = \frac{\sum_{S_j \in C_i} \mathrm{sim}(S_j, S_p)}{\sum_{S_m, S_n \in C_i} \mathrm{sim}(S_m, S_n)} \tag{9}$$

In (9), $S_p$ is the session that being merged into, $S_j$, $S_m$, $S_n \in C_i$, $1 \le i \le k$, $1 \le j, m, n \le n_i$, k is the number of clusters. $n_i$ is the number of Web sessions in $C_i$. Eq. (1) shows that $S_p$ should be merged into the cluster which make E be maximum, i.e. $E_r = \max \{ E_1, E_2, \ldots, E_k \}$, $S_p \in C_r$.

- Determination of initial data points

The accuracy of initial data points in clusters will directly affect the results of clustering with different qualities. Generally, for a clustering algorithm, which cluster a data point will be merged into is determined by the relationship (a criterion function) of the current data point and the data points remained in clusters. If the initial data point is inaccurate, the merger of current data point may be false. However, most clustering algorithm, such as k-means, ROCK, etc, just select k data points from data set randomly as the initial clusters. This may affect the results of clustering. For example, if an outlier is selected, the clusters are figured out according to the outlier may be very different to the desired clusters. Although k-means and k-medoids update centroid and representation during the process of clustering, the quality of finally results will be affected by the initial inaccurate data points, even they usually merge a data point into other cluster which should have been merged into cluster $C_i$.

Obviously, k data points selected from data set randomly can't represent the k initial clusters. The initial data points in clusters should be computed by a sample which drawn from data set according to theorem of sample. On the other hand, ROCK can yield satisfactory results for numeric attributes as well as categorical attributes. The shortcoming of ROCK is it's time and space complexity. So, it's feasible to use ROCK to determine initial Web session clusters.

### *B. Algorithm of Web clustering based on increase of similarities*

As analyzed above, we propose a new Web session clustering algorithm, i.e. WSCBIS (Web Sessions Clustering Based on Increase of Similarities) in this paper. This algorithm defines the number of clusters according to the Web site's structures, Web site's contents and the kinds of user's interesting behaviors which the analyzer desired. It takes advantage of ROCK to decide the initial points of each cluster and determines the criterion function according to the contributions of overall increase in similarities made by dividing Web sessions into different clusters. The process of the WSCBIS is presented in Fig. 10.

Statements before the first for-loop are operated on the basis on sample, so the execution time is insignificant. The time complexity is determined by the second for-loop mainly. As for the second for-loop, the time complexity of the first for-loop is $O(k \times |C_m|)$ where $|C_m|$ is the number of sessions in cluster $C_m$ and the time of the

second for-loop is O(k). So WSCBIS algorithm has a worst-case time complexity of $O(n \times k \times |C_{max}|)$ where n is the number of sessions in Web session file, k is the number of desired clusters and $|C_{max}|$ is the maximum number of Web sessions. WSCBIS needs two arrays to storage the overall similarities of each cluster and the similarities between the current session and the cluster respectively. Both the lengths are k. Thus the space complexity of WSCBIS is O(n), where n is the number of sessions in Web session file.

```
Input: session file, matrix of session similarity, number
of session clusters
Output: clusters
WSCBIS（SessionFile, k）{
  Draw sample S from SessionFile;
  ROCK（S,k）;
  for(m=1;m<=k;m++) SS[m]=∑_{Si,Sj∈Cm} sim(Si,Sj);
  for each Sp∈SessionFile{
    for(m=1;m<=k;m++)
      for(q=1;q<=|Cm|;q++)
        SI[m] += sim(Sp,Sq);
    E=0;
    for(m=1;m<=k;m++)
      if(SI[m]/SS[m]>E){
        E=SI[m]/SS[m];
        i=m;
      }
    Ci=Ci∪Sp;
    SS[i]=SS[i]+SI[i];
    }
}
```

Figure 10. WSCBIS algorithm

## VI. EXPERIMENTS

### A. Data Preprocessing

To validate the effectiveness and efficiency of our methodology mentioned above, we have made an experiment with the web server log of the library of South-Central University for Nationalities. The initial data source of our experiment is from May 28, 2006 to June 3, 2006, which size is 129MB. Our experiments were performed on a 2.8GHz Pentium Ⅳ CPU, 512MB of main memory, Windows 2000 professional, SQL Server 2000 and JDK 1.5.

As shown in Table 1, after data cleaning, the number of requests declined from 747890 to 112783. Fig. 11 shows the detail changes in data cleaning.

TABLE I. THE PROCESSES AND RESULTS OF DATA PREPROCESSING IN WEB SESSION CLUSTERING

| Entries in raw Web | Entries after data cleaning | Number of users | Number of sessions |
|---|---|---|---|
| 747 890 | 112 783 | 55 052 | 57 245 |

In Fig. 11, Bar chart 1 represents the initial requests in raw web log. From Bar chart 2 to 6 represent the requests after removing the log entries with filename suffix “gif” or “GIF”, the log entries with filename suffix “jpg” or “jpeg”, the log entries with filename suffix “css”, robots’ requests and error’s requests.

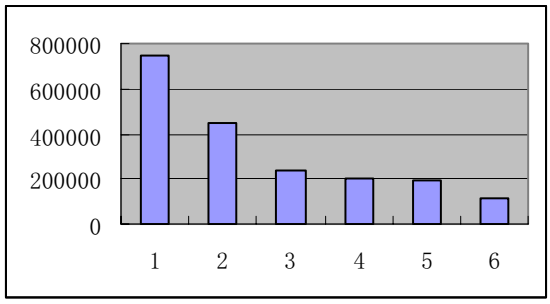


Figure 11. Processes of data cleaning

Fig. 12 is the processes of user identification. Bar chart 1 is the number of users identified only by IP addresses. Bar chart 2 is the number of users with the same IP address and agent. Bar chart 3 is the number of users considering local cache and proxy server. The values of Bar chart 1, 2 and 3 are 4575, 5440 and 55052 respective. The results of user identification accord with the facts because we set many proxy servers actually in our university.

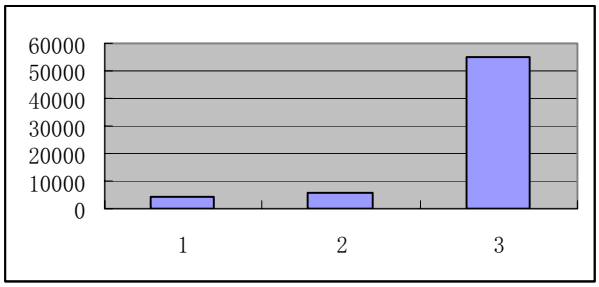


Figure 12. Processes of user identification

Finally, on the basis of user identification’s results, we have identified 57245 sessions by a threshold of 30 minutes and path completion.

### B. Algorithm of Web Session Clustering

In our experiments, we have applied k-means, ROCK and WSCBIS to the final results of Table I. Fig. 13 shows the execution time of k-means, ROCK and WSCBIS. We selected 1000-8000 sessions from session set as the input of three algorithms, where we draw initial clusters randomly both in k-means and in ROCK. In WSCBIS, we draw 150 sessions randomly from session set and using ROCK to compute initial clusters. When computing link in ROCK, sessions will be considered neighbors if their similarity is greater than $0.65(\theta=0.65)$, $f(\theta)=\frac{1-\theta}{1+\theta}=0.212$.

From Fig. 13 we can see that the time complexity of WSCBIS and k-means are linear, and they spend almost the same time for the same data source. However, the time complexity of ROCK is square, and the time spending is rising rapidly with data points’ increase.

Fig. 14 shows the quality of 3 algorithms’ clustering results. Experimental data sets are from similarity matrix consists of 57245 sessions. According to the structure of

Web site, we have extracted 10 clusters. Thereby k that is needed to enter the algorithm is 10. For k-means algorithm, centroid is the means of Web sessions' similarities in clusters. The summation of variance of current session and each cluster's centroid is as the criterion function. For ROCK algorithm, due to larger volume of data, we followed [19]. For WSCBIS algorithm, we first drew 500 sessions as the sample, and then applied ROCK algorithm to the sample to get 10 initial clusters. In order to get the clusters, we applied (9) to the remaining sessions.

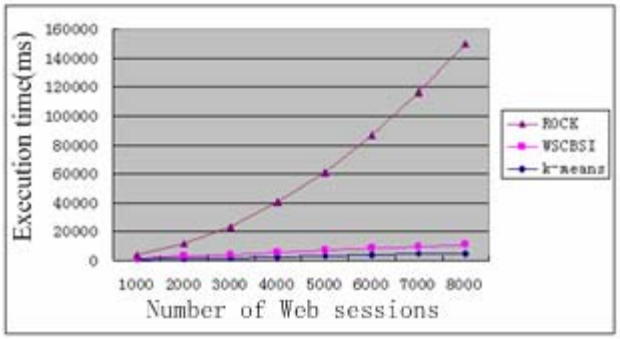


Figure 13. Comparison of 3 algorithm's execution time

We can see from Fig. 14 that the result of WSCBIS clustering algorithm is similar to ROCK, while it is greater different to k-means. It is difficult to evaluate the quality of a clustering's result. But it is well known that k-means predispose to clusters of proper convex shapes and is suitable for small or medium-sized data sets only. ROCK can find clusters of different shapes and has good scalability. ROCK can work at both numerical and categorical attributes. All these illuminate that the result of WSCBIS should be accurate.

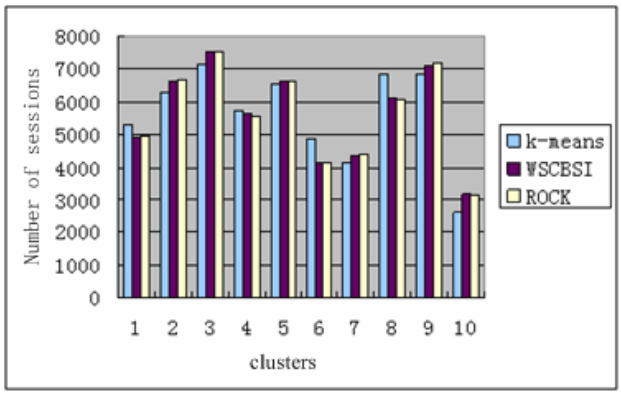


Figure 14. Comparison of 3 clustering results

## VII. CONCLUSION

Web clustering is a useful technique for grouping web sessions such that sessions within a single group/cluster have similar characteristics (or are close to each other), while sessions in different groups are dissimilar.

This paper has attempted to provide an up-to-date survey of the rapidly growing area of Web session clustering. We proposed a framework of Web session clustering. Next, we discussed the process of Web session clustering according to our framework in detail.

Data preprocessing is necessary for performing Web session clustering. We give some rules in every phase of data preprocessing in order to design and implement them easily. They not only reduce log file size but also increase the quality of the available data.

How to measure the similarity between web sessions is very important for web clustering. In this paper, we have analyzed the shortcoming of traditional similarity measurement between web sessions. Then we proposed the method to measure the similarity of web page access, and according to this similarity measurement of web page access, we introduced a new method to measure the similarity of web sessions using sequence alignment in computational biology.

Finally, we pointed out that number of clusters, the initial point of the respective clusters, and the defining of criterion function are the 3 key points and difficulties that deserve consideration in Web session clustering. Then we proposed a novel Web session clustering algorithm named WSCBIS. Our experiments have shown that its speed of execution is close to k-means and its quality of clusters is hard upon ROCK.

## REFERENCES

[1] W. Wang and O. R. Zaane. "Clustering Web sessions by sequence alignment," Proceedings of the 13th International Workshop on Database and Expert Systems Applications. Washington,DC:IEEE Computer Society, 2002,pp.394-398

[2] P. Berkhin. "Survey of clustering data mining techniques," Springer Berlin Heidelberg, Berlin,2006

[3] C. F. Li . "Data source analysis on Web session clustering," Journal of south-central university for nationalities, 2005(4), pp. 82-85

[4] E. Cohen, B. Krishnamurthy and J. Rexford. "Improving end-to-end performance of the web using server volumes and proxy filters," In Proc. ACM SIGCOMM, 1998, pp. 241~253

[5] D. Tanasa and B. Trousse. "Advanced data preprocessing for intersites Web session clustering," Intelligent Systems, IEEE, 2004(19), pp. 59 – 65

[6] L. Catledge and J. Pitkow. "Characterizing browsing behaviors on the World Wide Web," Computer Networks and ISDN Systems ,1995, vol. 27, no. 6, pp. 1065-1073

[7] M. S. Chen, J. S. Park and P. S. Yu. "Data mining for path traversal patterns in a web environment," In Proceedings of the 16th International Conference on Distributed Computing Systems, 1996, pp. 385-392

[8] C. Shahabi, A. Zarkesh and J. Adibi. "Knowledge discovery from users' web-page navigation," Proceedings of the 7th International Workshop on Research Issues in Data Engineering (RIDE '97) High Performance Database Management for Large-Scale Applications, Washington, DC, USA, IEEE Computer Society, 1997, pp. 20-31

[9] Y. Fu, K. Sandhu and M. Y. Shih. "Clustering of Web Users Based on Access Patterns," Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining Workshop on Web Mining, San Diego, CA: Springer-Verlag , 1999, pp. 560-567

[10] B. Mobasher, R. Cooley and J. Srivastara. "Automatic personalization based on Web session clustering," Communications of ACM, 2000, vol. 43, no. 8, pp. 142-151

[11] G. A. Banerjee. "Clickstream clustering using weighted longest common subsequences," Proc of the Workshop on

Web Mining, SIAM Conference on Data Mining. Chicago,USA,2001, pp. 158-172

[12] K. Charter, J. Schaeffer and D. Szafron. "Sequence alignment using FastLSA," Proceedings of METMBS'2000, Las Vegas: Nevada, 2000, pp. 239-245

[13] D. Hirschberg. "A linear space algorithm for computing maximal common subexpressions," Communications of the ACM, 1975, vol. 18, no. 6, pp.341-343

[14] K. Li, L.J. Cui. "Study of clustering algorithm based on model data," 2007 International Conference on Machine Learning and Cybernetics, 2007, pp.3961–3964

[15] J. Yu . "General C-means clustering model". IEEE Transactions on Pattern Analysis and Machine Intelligence, 2005, Vol. 27, No. 8, pp.1197-1211

[16] W. C. Li, Y. Zhou and S. X. Xia. "A novel clustering algorithm based on hierarchical and K-means clustering," Proceedings of the 26th Chinese Control Conference, 2007, pp.605-609

[17] J. J. Wu, H. Xiong, J. Chen and W. J. Zhou. "A generalization of proximity functions for K-means," 2007 Seventh IEEE International Conference on Data Mining, 2007, pp.361-370

[18] X. Q. Chen, H. Peng, J. S. Hu. "K-medoids substitution clustering method and a new clustering validity index method," The Sixth World Congress on Intelligent Control and Automation, 2006, pp.5896–5900

[19] S. Guha, R. Rastogi, K. Shim. "ROCK: a robust clustering algorithm for categorical attributes," Information Systems, 2000, Vol. 25, No. 5, pp.345-366

**Li Chaofeng** was born in 1974 in Henan Province China. He received the bachelor degree in computer science from the South-Central University for Nationalities, Wuhan, China, in 1997, and the M.S. and Ph.D. degrees in computer science from the Huazhong University of Science and Technology, Wuhan, China, in 2001 and 2007, respectively.

Since 1997 he has been on the faculty of the College of Management, South-Central University for Nationalities, where he is currently an Associate Professor. He has published about 20 papers in refereed journals and conferences in the areas of databases, artificial intelligence, and data mining. His current research is in the areas of databases, distributed systems, and web usage mining.