Automatic Web User Profiling and Personalization Using Robust Fuzzy Relational Clustering

Olfa Nasraoui[1], Raghu Krishnapuram[2], Anupam Joshi[3], and Tapan Kamdar[3]

[1] Department of Electrical and Computer Engineering,
The University of Memphis, Memphis TN 38152, USA,
[2] IBM India Research Lab, Block 1, Indian Institute of Technology,
Hauz Khas, New Delhi 110016, India
[3] Department of Computer Science and Electrical Engineering
University of Maryland - Baltimore County, Baltimore, MD 21250, USA

Abstract. The proliferation of information on the world wide Web has made the personalization of this information space a necessity. Personalization of content returned from a Web site is a desired feature that can enhance server performance improve system design, and lead to wise marketing decisions in electronic commerce. Mining typical user profiles from the vast amount of historical data stored in access logs is an important component of Web personalization. In the absence of *a priori* knowledge, unsupervised or clustering methods seem to be ideally suited to categorize the usage behavior of Web surfers. In this chapter, we present a framework for mining typical user profiles from server acces logs based on robust fuzzy relational clustering. As a by-product of the clustering process that generates robust profiles, associations between different URL addresses on a given site can easily be inferred. In general, the URLs that are present in the same profile tend to be visited together in the same session or form a large *itemset*. Finally, we present a personalization system that uses previously mined profiles to automatically generate a Web page containing URLs the user might be interested in. Our personalization approach is based on profiles computed from the prior traversal patterns of the users on the website and do not involve providing any declarative private information or the user to log in.

1 Introduction

One of the data repositories, affecting every aspect of our life lately, is the World Wide Web. In addition to its ever-expanding size and lack of structure, the WWW has not been responsive to user preferences and interests. A user looking for some specific information has to wade through a morass of information, get bombarded with irrelevant information, and often lose track of their initial objective. One way to deal with this problem is through personalization. Personalization can either be done via information brokers (e.g. Web search engines), or in an *end-to-end* manner by making Web sites adaptive. Initial work in this area has basically focused on creating recommender

systems. The Firefly system [41] attempted to provide CDs that best match a user's professed interests. The Webwatcher project [2] at CMU highlights hyperlinks in a page based on the declared interests and the path traversal of a user as well as the path traversals of previous users with similar interests. W^3IQ [18, 19] and PHOAKS [44] have sought to use cooperative information retrieval techniques for personalization. "Mining" information from the user's interaction is another approach towards personalization. Perkowitz and Etzioni [33, 34] proposed adapting Web pages based on a user's traversal pattern. The standard K-Means algorithm was used to cluster users's traversal paths in [40]. In [6], associations and sequential patterns between web transactions are discovered based on the Apriori algorithm [1]. There is also a recent body of work [3] which seeks to transform the Web into a more structured, database like entity and then using standard OLAP techniques on it [32]. It is important to mention that most of the above efforts have relied on relatively simple techniques which can be inadequate for real user profile data since they are neither resilient to the "noise" typically found in user traversal patterns, nor able to handle the uncertainties and fuzziness inherent in Web data. To deal with the fuzzy nature of Web data and to automatically determine the number of clusters, Nasraoui et al. [24, 29] proposed to extract profiles using an unsupervised relational clustering algorithm based on the competitive agglomeration algorithm [10]. In their work, they have also proposed formal definitions for the Web user profile as well as quantitative evaluation measures. Also, to handle possibly unknown noise contamination rates in Web data, Nasraoui et al [28] have proposed mining the Web log data using a fuzzy relational clustering algorithm based on a robust estimator. In this work, they have also extended the formal definition of a Web user profile and its quantitative evaluation measures to a "robust" user profile and "robust" evaluation measures.

A related topic that has been recently gaining momentum is the idea that we can learn much about users and customers by tracking and analyzing their *clickstreams*, which is of great importance in e-commerce.

In the absence of any *a priori* knowledge, unsupervised classification or clustering methods seem to be ideally suited to analyze the semi-structured log data of user accesses by categorizing them into automatically generated classes of user session profiles. We define the notion of a "user session" as being a temporally compact sequence of web accesses by a user. The goal of our Web mining is to categorize these sessions. In this light, Web mining can be viewed as a special case of the more general problem of knowledge discovery in databases [9]. We define a new distance measure between two Web sessions that captures the organization of a Web site. This organizational information is inferred directly from the URLs.

Web mining involves data that is mildly to severely corrupted with "noise". Outliers and incomplete data can easily occur in the data set due to a wide variety of reasons inherent to Web browsing and logging. Moreover, the noise

contamination rate and the scale of the data is rarely known in advance. For example, consider the situation where we are analyzing log entries to discover typical information access patterns. Clearly, there is a significant percentage of time (sometimes as large as 20-30 percent) that a user is simply "browsing" the Web and does not follow any particular pattern.

Categories in most data mining tasks are rarely well separated. The class partition is best described by fuzzy memberships [5], particularly along the overlapping borders. Specifically, in Web mining, certain access patterns may be regarded as belonging to more than one class with different degrees instead of belonging to only one class. Also, at times, it will be necessary to work with relational[1] data because it is intuitively and conceptually easier to describe the relation or similarity between two objects than to map them to numerical features. In fact, determining the nature and number of features is itself very challenging. This problem is particularly accute when the data contains nonnumeric fields. This means that relational clustering methods are expected to be highly beneficial in Web mining. As will be explained later, Web sessions are too complex to convert to simple numerical features. Hence, clustering the user sessions could be tackled by exploiting inter-session similarities within a relational framework.

For all these reasons, we present robust fuzzy relational methods to cluster the user sessions based on their pair-wise dissimilarities, and illustrate their use in extracting user profiles from real log data. The Relational Fuzzy $C-$ Maximal Density Estimator (RF$C-$MDE) [30] and the Robust Fuzzy C Medoids (FCMdd) can deal with complex and subjective distance/similarity measures which are not restricted to be Euclidean. The web sites for the department of Computer Engineering and Computer Sciences at the University of Missouri, as well as that of the department of Computer Sciences and Electrical Engineering at the University of Maryland-Baltimore County were used as testbeds for our algorithms which successfully analyzed server access logs and obtained typical session profiles of users. The performance of our techniques was superior to that of the Non Euclidean Relational Fuzzy C-Means (NERF) [16].

We also note that as a by-product of our clustering process, associations between different URL addresses on a given site can easily be inferred from the resulting robust profiles. In general, the URLs that are present in the same profile tend to be visited together in the same session, i. e., they form a large item set. These item sets are similar to the ones computed by the Apriori algorithm [1], and have been applied to mining associations and sequential patterns between web transactions in [6]. However, unlike the latter, our method mines typical users profiles as a primary goal, and the discovered associations resulting from this profile extraction come with no additional computation. The biggest advantage of our approach is that clustering results do not depend on the goodness or completeness of any pre-discovered

[1] Note that this term is used in its statistical sense, not in its database sense

associations. Some of these associations (ones with low support) can be very hard and time consuming to discover from Web session data.

The rest of the chapter is organized as follows. In Section 2, we present a framework for Web usage mining based on clustering the Web user sessions. In Section 3, we present an overview of relational clustering algorithms. In Section 3.1, we present the Relational Fuzzy C Maximal Density Estimator (RF$C-$MDE), and illustrate its performance in extracting robust session profiles from the access log files of several real Web sites. In Section 4 we present the Robust Fuzzy C Medoids (FCMdd) algorithm as well as a low-complexity version of it, and use it to extract typical session profiles from real access log files. Finally, in Section 6, we present the conclusions.

2 The Knowledge Discovery Process of Web Session Profiling

2.1 Extracting Web User Sessions

The access log for a given Web server consists of a record of all files accessed by users. Each log entry consists of: (i) User's IP address, (ii) Access time, (iii) URL of the page accessed, $\cdots$, etc. A user session consists of accesses originating from the same IP address within a predefined time period. Each URL in the site is assigned a unique number $j \in \{1, \ldots, N_U\}$, where N_U is the total number of valid URLs. Thus, the i^{th} user session is encoded as an N_U-dimensional binary attribute vector $\mathbf{s}^{(i)}$ with the property

$$s_j^{(i)} = \begin{cases} 1 \text{ if the user accessed the } j^{th} \text{ URL during the } i^{th} \text{ session} \\ 0 \text{ otherwise} \end{cases}$$

The ensemble of all N_S sessions extracted from the server log file is denoted $\mathcal{S}$.

2.2 Assessing Web User Session Similarity

The similarity measure between two user-sessions: $\mathbf{s}^{(k)}$ and $\mathbf{s}^{(l)}$ relies on two sub-measures [29, 30]. The first measure which ignores the site structure is given by $S_{1,kl} = \frac{\sum_{i=1}^{Nu} s_i^{(k)} s_i^{(l)}}{\sqrt{\sum_{i=1}^{Nu} s_i^{(k)}} \sqrt{\sum_{i=1}^{Nu} s_i^{(l)}}}$. The second similarity measure requires the pre-computation of the similarities at the structural URL level that will be used in the computation of the similarity at the session level.

The entire Web site is modeled as a tree with the nodes representing different URL's. The tree is similar to that of a directory where an edge connects one node to another if the URL corresponding to the latter is hierarchically located under that of the former, The "syntactic" similarity between the i^{th} and j^{th} URLs is defined as $S_u(i,j) = \min\left(1, \frac{|p_i \cap p_j|}{\max(1, \max(|p_i|, |p_j|) - 1)}\right)$, where p_i denotes the path traversed from the root node (main page) to the node corresponding to the i^{th} URL, and $|p_i|$ indicates the length of this path.

Note that this similarity which lies in $[0,1]$ basically measures the amount of overlap between the paths of the two URLs. This overlap is inferred directly from the URL address string by exploiting the one-to-one mapping between the address and the site topology. The pairwise URL similarities should be computed only once offline for a particular Web site prior to any clustering. Now the similarity on the session level which incorporates the syntactic URL similarities is computed by $S_{2,kl} = \frac{\sum_{i=1}^{N_U}\sum_{j=1}^{N_U} s_i^{(k)} s_j^{(l)} S_u(i,j)}{\sum_{i=1}^{N_U} s_i^{(k)} \sum_{j=1}^{N_U} s_j^{(l)}}$. The final similarity given by a maximally optimisitc aggregation of $S_{1,kl}$ and $S_{2,kl}$ is $S_{kl} = \max(S_{1,kl}, S_{2,kl})$. Finally, this similarity is mapped to the dissimilarity measure $d_s^2(k,l) = (1 - S_{kl})^2$. This dissimilarity measure satisfies the desirable properties: $d_s^2(k,k) = 0$, $d_s^2(k,l) >= 0\ \forall k,l$, and $d_s^2(k,l) = d_s^2(l,k)\ \forall k,l$. However, unlike a metric distance it violates the triangular inequality in some cases. For instance, the dissimilarity between the sessions {/courses/cecs345/syllabus} and {/courses/cecs345} is zero. So is the dissimilarity between {/courses/cecs345} and {/courses/cecs401}. However, the dissimilarity between {/courses/cecs345/syllabus} and {/courses/cecs401} is not zero (it is 1/4). This illustrates a desirable property for profiling sessions which is that the dissimilarity becomes more stringent as the accessed URLs get farther from the root because the amount of specificity in user interest increases correspondingly.

2.3 Clustering Web User Sessions

After pre-computing all pairwise dissimilarities, the extracted sessions can be clustered using relational clustering. We will present two robust fuzzy relational clustering techniques that can handle the noisy and fuzzy nature of Web usage data in Sections 3.1 and 4.

2.4 Interpretation and Evaluation of the Results

The results of clustering the user session data are interpreted using the following quantitative measures [30]. First, the user sessions are assigned to the closest clusters based on the computed distances, d_{ik}, from the i^{th} cluster to the k^{th} session. This creates C clusters $\mathcal{X}_i = \left\{ \mathbf{s}^{(k)} \in \mathcal{S} \mid d_{ik} < d_{jk}\ \forall j \neq i \right\}$, for $1 \leq i \leq C$.

The sessions in cluster $\mathcal{X}_i$ are summarized by a typical session "profile" vector [30] $\mathbf{P}_i = \left(P_{i1}, \ldots, P_{i_{N_U}} \right)^t$. The components of $\mathbf{P}_i$ are URL relevence weights, estimated by the probability of access of each URL during the sessions of $\mathcal{X}_i$ as follows

$$P_{ij} = p\left(\mathbf{s}_j^{(k)} = 1 | \mathbf{s}_j^{(k)} \in \mathcal{X}_i \right) = \frac{\left| \mathcal{X}_{i_j} \right|}{|\mathcal{X}_i|}, \quad (1)$$

where $\mathcal{X}_{i_j} = \left\{ \mathbf{s}^{(k)} \in \mathcal{X}_i \mid s_j^{(k)} > 0 \right\}$. The URL weights P_{ij} measure the significance of a given URL to the i^{th} profile. Besides summarizing profiles, the components of the profile vector can be used to recognize an invalid profile which has no strong or frequent access pattern. For such a profile, all the URL weights will be low.

Several classical cluster validity measures can be used to assess the goodness of the partition. The intra-cluster or within-cluster distance represents an average of the distances between all pairs of sessions within the i^{th} cluster, and is given by $\overline{D}_{Wi} = \frac{\sum_{\mathbf{s}^{(k)} \in \mathcal{X}_i} \sum_{\mathbf{s}^{(l)} \in \mathcal{X}_i, l \neq k} d_{kl}^2}{|\mathcal{X}_i|(|\mathcal{X}_i|-1)}$. This is inversely related to the compactness or goodness of a cluster. A good guideline to use when evaluating clusters based on the intra-cluster distances is to compare these values to the total average pairwise distance of all sessions. The latter corresponds to the intra-cluster distance if all the user sessions were assigned to one cluster (i.e., no category information is used). Also it is important to recall that all distances are in $[0, 1]$. The inter-cluster or between-cluster distance represents an average of the distances between sessions from the i^{th} cluster and sessions from the j^{th} cluster, and is given by $\overline{D}_{Bij} = \frac{\sum_{\mathbf{s}^{(k)} \in \mathcal{X}_i} \sum_{\mathbf{s}^{(l)} \in \mathcal{X}_j} d_{kl}^2}{|\mathcal{X}_i||\mathcal{X}_j|}$. For a good partition, the inter-cluster distances should be high because they measure the separation between clusters.

3 Relational Clustering

The term "relational data" refers to the situation where we have only numerical values representing the degrees of similarity or relation between the pairs of objects in the data set. In contrast, "object data" refers to the the situation where the objects to be clustered are explicitly represented by vectors $\mathbf{x}_i \in \Re^p$. Algorithms that generate partitions of relational data are usually referred to as relational clustering algorithms. Relational clustering is more general in the sense that it is applicable to situations in which the objects to be clustered cannot be represented by numerical features. For example, we can use relational clustering algorithms to cluster URLs (Universal Resource Locators) if we can define a dissimilarity measure to quantify the degree of resemblance between pairs of URLs. The pair-wise dissimilarities are usually stored in the form of a matrix called the dissimilarity matrix.

One of the most popular relational clustering algorithms is the SAHN (Sequential Agglomerative Hierarchical Non-overlapping) model [42] which is a bottom-up approach that generates crisp clusters by sequentially merging pairs of clusters that are closest to each other in each step. Depending on how "closeness" between clusters is defined, the SAHN model gives rise to single, complete and average linkage algorithms. A variation of this algorithm can be found in [13]. Another well-known relational clustering algorithm is PAM (Partitioning Around Medoids) due to Kaufman and Rousseeuw [21].

This algorithm is based on finding k representative objects (also known as *medoids* [20]) from the data set in such a way that the sum of the within cluster dissimilarities is minimized. A modified version of PAM called CLARA (Clustering LARge Applications) to handle large data sets was also proposed by Kaufman and Rousseeuw [21]. Ng and Han [31] propose another variation of CLARA called CLARANS. This algorithm tries to make the search for the k representative objects (medoids) more efficient by considering candidate sets of k medoids in the neighborhood of the current set of k medoids. However, CLARANS is not designed for relational data. Finally, it is also interesting to note that Fu [11] suggested a technique very similar to the k medoid technique in the context of clustering string patterns generated by grammars in syntactic pattern recognition. Some of the more recent algorithms for relational clustering include [12], [35], [43], and [4].

SAHN, PAM, CLARA and CLARANS generate crisp clusters. When the clusters are not well defined (i.e., when they overlap) we may desire fuzzy clusters. Two of the early fuzzy relational clustering algorithms are the ones due to Ruspini [39] and Diday [8]. Other notable algorithms include Roubens' Fuzzy Non Metric Model or FNM [36], Windham's Association Prototype Model or AP [45], Hathaway & Bezdek's Relational Fuzzy c-Means [17], and Kaufman & Rousseeuw's Fuzzy Analysis or FANNY [21]. Recently, Nasraoui et al. [24, 29] presented an unsupervised relational clustering algorithm based on the competitive agglomeration algorithm [10].

3.1 The Relational Fuzzy $C-$ Maximal Density Estimator

It is known that for complex data sets containing overlapping clusters, fuzzy partitions model the data better than their crisp counterparts. In particular, fuzzy memberships are richer than crisp memberships in describing the degrees of belonging of data points lying in the areas of overlap. Moreover, fuzzy partitions generally make the optimization process less prone to local or suboptimal solutions. Let $\mathcal{X} = \{\mathbf{x}_j \mid j = 1, \ldots, N\}$ be a set of feature vectors in an $n-$dimensional feature space with coordinate axis labels $x_1,\ x_2,\ \ldots,\ x_n$, where $\mathbf{x}_j = (x_{j1},\ x_{j2},\ \ldots,\ x_{jn})$. Let θ represent the parameters to be estimated. With a fuzzy partition, a data point $\mathbf{x}_j$ belongs to each cluster, $\mathcal{X}_i$, to a varying degree called fuzzy membership u_{ij}. A fuzzy partiton must satisfy the following constraints [5],

$$0 \leq u_{ij} \leq 1$$

$$\sum_{i=1}^{C} u_{ij} = 1 \ \forall j = 1 \cdots N$$

$$0 \leq \sum_{j=1}^{N} u_{ij} \leq N \ \forall i = 1 \cdots C$$

Using fuzzy memberships offers other advantages in terms of smoothing the surface of the objective function and reducing its discontinuity [25], thus making the problem analytically more tractable. The Maximal Density Estimator (MDE) [26] is a new robust estimator that is free of any presuppositions about the noise proportion. One way to use a fuzzy partition while estimating the cluster parameters with the MDE is to use the following criterion

$$\min_{\mathbf{\Theta},\sigma_i,u_{ij}} \quad \left\{ J = \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij}^m w_{ij} \frac{d_{ij}^2}{\sigma_i} - \alpha \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij}^m w_{ij} \right\}, \tag{2}$$

where $m \in [1, \infty)$ is a fuzzifier parameter that controls the degree of fuzziness of the resulting partition, with values closer to 1 corresponding to crisper partitions, and w_{ij} is given by

$$w_{ij} = \exp \frac{-d_{ij}^2}{2\sigma_i}. \tag{3}$$

In (2), d_{ij}^2 is the residual of data point $\mathbf{x}_j$ with respect to the fit computed from the i^{th} cluster prototype θ_i, and w_{ij} is a positive weight associated with point $\mathbf{x}_j$. The weight w_{ij} can be considered as the degree of membership of data point $\mathbf{x}_j$ in the inlier set or the set of good points associated with the i^{th} cluster. The first term of this objective function tries to minimize the scaled residuals of the good points. The second term of this objective function tries to use as many good points (inliers) as possible in the estimation process, via their high weights. Thus the combined effect is to optimize the density, i.e., the ratio of the total number of good points to the volume. We choose the value of α to balance the two terms. In 2-D, we choose $\alpha = 1$. For the general n-dimensional case α should be close to n, since the ratio of the first term to the second term approaches the average of a χ^2 distribution for Gaussian data. Finally, we should note that d_j^2 should be a suitable distance measure, tailored to detect desired shapes, such as the Euclidean distance for spherical clusters, or the Gustafson-Kessel (GK) distance [14] for ellipsoidal clusters characterized by a covariance matrix, etc.

Since each cluster is independent of the rest, it is easy to derive the optimal update equations for the parameters for each cluster. The scale parameter of the i^{th} cluster is given by

$$\sigma_i = \frac{1}{(2+\alpha)} \frac{\sum_{j=1}^{N} u_{ij}^m w_{ij} d_{ij}^4}{\sum_{j=1}^{N} u_{ij}^m w_{ij} d_{ij}^2}. \tag{4}$$

To find the optimal prototype parameters θ_i of the i^{th} cluster, we set

$$\frac{\partial J}{\partial \theta_i} = \frac{1}{\sigma_i} \quad \sum_{\mathbf{x}_j \in \mathcal{X}_i} u_{ij}^m w_{ij} \frac{\partial d_{ij}^2}{\partial \theta_i} = \mathbf{0}.$$

For instance for the case of location estimation, d_{ij}^2 is the squared Euclidean distance $d_{ij}^2 = \|\mathbf{x}_j - \mathbf{c}_i\|^2$, and the center $\mathbf{c}_i$ is given by

$$\mathbf{c}_i = \frac{\sum_{j=1}^{N} u_{ij}^m w_{ij} \mathbf{x}_j}{\sum_{j=1}^{N} u_{ij}^m w_{ij}} \tag{5}$$

To simplify the algorithm, we de-couple the robust parameter estimation process via the robust weights, w_{ij}, from the partitionning process, via the memberships, u_{ij}. Since the parameter estimates in (5) which optimize (2) are already robust, we use the simpler Fuzzy c-Means [5, 39] objective function to update the memberships u_{ij} in each iteration as follows:

$$u_{ij} = \frac{\left(\frac{1}{d_{ij}^2}\right)^{\frac{1}{m-1}}}{\sum_{k=1}^{C} \left(\frac{1}{d_{kj}^2}\right)^{\frac{1}{m-1}}}. \tag{6}$$

Therefore, the optimization process will consist of alternating updates of the memberships, as given by (6), and the cluster centers and scale parameters as given by (5) and (4). We call the resulting clustering algorithm the Fuzzy $C-$ Maximal Density Estimator (F$C-$MDE).

The F$C-$MDE algorithm was originally formulated to deal with object or feature data only. For relational applications, F$C-$MDE must be extended so that it can work on relational data. First, the squared Euclidean distance, $d_{ik}^2 = \|\mathbf{x}_k - \mathbf{c}_i\|^2$, from feature vector $\mathbf{x}_k$ to the center of the i^{th} cluster, $\mathbf{c}_i$, is written in terms of the relation matrix $\mathbf{R}$ as follows [16]:

$$d_{ik}^2 = (\mathbf{R}\mathbf{v}_i)_k - \mathbf{v}_i^t \mathbf{R} \mathbf{v}_i / 2. \tag{7}$$

Here, $\mathbf{R} = [R_{jk}]$ is the dissimilarity between $\mathbf{x}_j$ and $\mathbf{x}_k$, and $\mathbf{v}_i$ is the membership vector defined by

$$\mathbf{v}_i = \frac{\left(u_{i1}^m w_{i1}, \ldots, u_{iN}^m w_{iN}\right)^t}{\sum_{j=1}^{N} u_{ij}^m w_{ij}}. \tag{8}$$

Equation (7) allows the computation of the distance between the data points and cluster prototypes in each iteration when only the relational data, $\mathbf{R}$, are given. Therefore, a relational dual of F$C-$MDE exists for the special case where the object data and relational data satisfy

$$\mathbf{R} = [R_{jk}] = \|\mathbf{x}_j - \mathbf{x}_k\|^2 \tag{9}$$

This means that even when only relational data is available in the form of an $N \times N$ relation matrix, the relational dual of F$C-$MDE is expected to perform in an equivalent way to the F$C-$MDE provided that the relation matrix, $\mathbf{R}$, is Euclidean, i.e., there exists a set of N points in $\mathcal{R}^{N-1}$, called a realization of $\mathbf{R}$, satisfying (9).

When a realization does not exist for the relation matrix, $\mathbf{R}$, some of the distances computed using (7) may be negative. To overcome this problem, we use the β-spread transform [16] to convert a non-Euclidean matrix $\mathbf{R}$ into an Euclidean Matrix $\mathbf{R}_\beta$ as follows

$$\mathbf{R}_\beta = \mathbf{R} + \beta\,(\mathbf{M} - \mathbf{I}) \tag{10}$$

where β is a suitably chosen scalar, $\mathbf{I} \in \mathcal{R}^{n\times n}$ is the identity matrix and $\mathbf{M} \in \mathcal{R}^{n\times n}$ satisfies $M_{jj} = 1$ for $1 \le i, j \le n$. The distances d_{ik}^2 must be checked in every iteration for negativity, which indicates a non-Euclidean relation matrix. In that case, the β-spread transform should be applied with a suitable value of β to make the d_{ik}^2 positive again. An underestimate for the lower bound on β was derived [16] and related to the necessary shift that is needed to make the distances positive. This result can be summarized as

$$\Delta\beta = \max_{i,k}\{-2d_{ik}^2 / \left\|\mathbf{v}_j - \mathbf{e}_k\right\|^2\}, \tag{11}$$

where $\mathbf{e}_k$ denotes the k^{th} column of the identity matrix. An alternative approach [21] to this problem imposes Kuhn-Tucker conditions to ensure positivity of the memberships. The resulting Relational F$C-$MDE algorithm (RF$C-$MDE) can deal with complex and subjective dissimilarity/similarity measures which are not restricted to be Euclidean or metric. The RF$C-$MDE algorithm is summarized below:

The Relational Fuzzy C− Maximal Density Estimator (RFC−MDE)

Fix the number of clusters C;
Pick C distinct random rows from relation matrix to serve as the initial implicit prototypes, this results in initial distances d_{ik}^2 for $1 \le i \le C$;
Repeat
  Compute membership vectors $\mathbf{v}_i$ for $1 \le i \le C$ using (8);
  Compute $d_{ik}^2 = (\mathbf{R}_\beta \mathbf{v}_i)_k - \mathbf{v}_i^t \mathbf{R}_\beta \mathbf{v}_i/2$ for $1 \le i \le C$ and $1 \le k \le N_S$;
  If ($d_{ik}^2 < 0$ for any i and k) then {
    Compute $\Delta\beta$ by using (11);
    Update $d_{ik}^2 \leftarrow d_{ik}^2 + (\Delta\beta/2) * \left\|\mathbf{v}_j - \mathbf{e}_k\right\|^2$ for $1 \le i \le C$ and $1 \le k \le N_S$;
    Update $\beta = \beta + \Delta\beta$;
  }
  Compute robust weights w_{ik} for $1 \le i \le C$ and $1 \le k \le N_S$, using (3);
  Compute scale parameters σ_i for $1 \le i \le C$ using (4);
Until (*memberships stabilize*).

3.2 Experimental Results

The Web mining procedure described in Section 2 was used to extract typical user session profiles from the log data of the Web sites of two departments

at two different universities during 1998. We first describe the results for the department of Computer Engineering and Computer Sciences at the University of Missouri, Columbia. The log data from accesses to the server during a period of 12 days was used. After filtering out irrelevent entries, the data was segmented into 1703 sessions. The maximum elapsed time between two consecutive accesses in the same session was set to 45 minutes. The number of distinct URLs accessed in valid entries was 369. The initial distance values d_{ik}^2 were obtained by randomly choosing C rows, for $1 \leq i \leq C$, from the relation matrix and computing the fuzzy memberships using (6), which treats the transactions corresponding to the selected rows as initial prototypes.

After clustering the relational data with RFC−MDE and $C = 35$, the sessions were assigned to the clusters in a minimum distance classifier sense. As a result, only 15 clusters that had cardinalities exceeding 30 were kept, hence making a sufficiently strong profile. Table 1 illustrates four profiles computed using (1), where only the significant URLs ($P_{ij} \geq 0.15$) are displayed, and the individual components are displayed in the format $\{P_{ij}$ - j^{th} URL$\}$. The sessions were assigned to the closest cluster and the session clusters or profiles were examined qualitatively. The results are summarized in Table 2, which also lists the cardinality and the intra-cluster distance for all clusters.

Table 1. Examples of Profiles Discovered by RFC−MDE from Missouri Data

i	$\mathbf{P}_i$
4	{.78 - /cecs_computer.class} {.96 - /courses.html} {.99 - /courses_index.html} {.96 - /courses100.html} {.34 - /courses300.html} {.2 - /courses_webpg.html} {.28 - /courses200.html} {.87 - /}
7	{.47 - /cecs_computer.class} {0.85 - /degrees_undergrad.html} {0.84 - /degrees_index.html} {0.85 - /bsce.html} {0.46 - /bscs.html} {0.32 - /bacs.html} {0.31 - /courses.html} {0.31 - /courses_index.html} {0.31 - /courses100.html} {0.21 - /courses300.html} {0.18 - /courses200.html} {0.84 - /degrees.html} {0.22 - /general.html} {0.22 - /general_index.html} {0.22 - /facts.html} {0.16 - /research.html} {0.65 - /}
13	{.18 - /~shi} {.64 - /~shi/cecs345} {.38 - /~shi/cecs345/java_examples} {.19 - /~shi/cecs345/references.html} {.20 - /~shi/cecs345/Lectures/05.html} {.20 - /~shi/cecs345/Lectures/06.html} {.29 - /~shi/cecs345/Lectures/07.html} {.18 - /~shi/cecs345/Projects/1.html}

The results show that RFC−MDE succeeded in delineating many different profiles in the user sessions. Except for the 14^{th} cluster, all clusters correspond to real profiles reflecting distinct user interests. The profiles fol-

Table 2. Summary of RF$C-$MDE User Session Profiles for Missouri Data

i	$\lvert\mathcal{X}_i\rvert$	description	$\overline{D}_{W\,i}$
1	56	Professor 1's course pages	0.15
2	33	Access statistics pages	0.1
3	64	Site manager's pages	0.13
4	206	General course inquiries	0.2
5	94	main page, people, faculty, research and degree pages	0.68
6	141	cecs352 course pages	0.24
7	68	Undergraduate degree and general course inquiries	0.3
8	66	Accesses to adminstrator's pages	0.24
9	162	Accesses to the cecs227 class pages	0.25
10	62	Professor 2's main page and research papers	0.29
11	64	Sessions combining Dr Joshi's courses and research pages	0.66
12	191	Short sessions mostly limited to main page and class list	0.26
13	142	Professor 3's cecs345 general course enquiries	0.35
14	119	Mixture of unrelated accesses that don't make a strong profile	0.92
15	152	Dr. Saab's course pages	0.55

lowed the access patterns of typical users – the general "outside visitor" is captured in profiles 5 and 12, prospective students in profile 4 and 9, students in CECS352 in profile 6, etc. The goodness of these clusters is recognizable through their low intra-cluster distances (considerably lower than the total average pairwise session distance of 0.9), and their high inter-cluster distances (the majority between .9 and 1). Note that many sessions that do not belong to any profile are lumped in the 14^{th} profile which is easily recognized as a spurious cluster by using the quantitative evaluation measures. In fact, this particular cluster had no significant URLs ($P_{ij} < 0.15$ for all j) and its intra-cluster distance (0.92) was even higher than the total average pairwise distance of all sessions. In addition to the intra-cluster distance, the robust weights, w_{ij}, are extremely useful for extracting the core members of each class, as well as for distinguishing between strong and spurious profiles. In fact, only the core members of each cluster are expected to have a significant robust weight. For example when only sessions with weights exceeding 0.04 are considered, profiles Nos. 5, 14, and 15 end up having less than 10 members, hence making weak profiles. Note that the spurious cluster No. 14 was eliminated by this method as all its members have low robust weights. Also, several sessions were identified as noise with respect to certain profiles because of their low weights. For example, the following two sessions from profile No. 13, having low robust weights, were considered as noise:

{/courses400.html, /people_index.html, /~shi/publications, /~shi/cecs345, /~joshi/scipad}

{/~shi/cecs345, /~saab/cecs303/private, /~saab/cecs303/private/solution,

/~saab/cecs303/private/solution/hw1.html}

As can be seen, these sessions do not have a specific interest in the 13^{th} profile which concerns Dr Shi's pages.

The Non Euclidean Relational Fuzzy $C-$Means (NERF) [16] was also used to cluster the sessions relation matrix, and resulted in only 12 significant profiles, as shown in Table 3. RF$C-$MDE fared better than NERF in the sense that the spurious cluster corresponding to the 12^{th} cluster found by NERF (which corresponds to the 14^{th} profile found by RF$C-$MDE) had 339 more sessions. Also, NERF completely missed the clusters corresponding to the 11^{th} and 15^{th} profiles. Moreover, as can be inferred from the higher cardinality and the average intra-cluster distance for comparable profiles (such as the 5^{th} profiles found by RF$C-$MDE and NERF), NERF's clusters tend to contain more irrelevent sessions or noise. Unlike in the case of RF$C-$MDE, we cannot distinguish noise in the profiles obtained by NERF because it has no robust weights. This is because the fuzzy memberships, u_{ij}, are relative, and regardless of whether a data point is noisy, its memberships across all classes always sum to one, i.e., $\sum_{i=1}^{C} u_{ij} = 1 \ \forall j = 1 \cdots N$.

We also note that as a by-product of the clustering process, associations between different URL addresses on a given site can easily be inferred from the resulting robust profiles. In general, the URLs that are present in the same profile tend to be visited together in the same session. For example, by looking at the 7^{th} profile in Table 1, we can deduce that the URLs making up that profile tend to co-occur or form a large item set, similar to the ones computed by the Apriori algorithm [1]. Such item sets can be applied to mining associations and sequential patterns between web transactions in [6]. However, unlike the latter, our method mines typical users profiles as a primary goal, and the discovered associations resulting from this profile extraction come with no additional computation.

Table 3. Summary of NERF User Session Profiles for Missouri Data

i	$\lvert\mathcal{X}_i\rvert$	description	$\overline{D}_{Wi}$
1	70	Professor 1's pages	0.2
2	34	Access statistics pages	0.12
3	64	Site manager's pages	0.13
4	213	General course inquiries	0.21
5	109	Main page, people, research, and faculty pages	0.78
6	163	cecs352 course pages	0.28
7	69	Undergraduate degree and general course inquiries	0.31
8	66	Accesses to adminstrator's pages	0.24
9	172	Accesses to the cecs227 class pages	0.28
10	80	Professor 2's main page and research papers	0.35
11	206	Short sessions mostly limited to main page and class list	0.29
12	458	Mixture of unrelated accesses	0.87

The next experiment consists of extracting typical user session profiles from the log data of the Web site for the Computer Science department at the University of Maryland, Baltimore County during 1998. The log data from accesses to the server during a period of 6 hours in the late evening was used. After filtering out irrelevent entries, the data was segmented into 678 sessions. The maximum elapsed time between two consecutive accesses in the same session was set to 45 minutes. The number of distinct URLs accessed in valid entries was 1681.

After clustering the relational data with RF$C-$MDE and $C = 35$, the sessions were assigned to the clusters in a minimum distance classifier sense. As a result, only 12 clusters that had cardinalities exceeding 10 were kept, i.e., those making a sufficiently strong profile. Table 4 illustrates four profiles computed using (1), where only the significant URLs ($P_{ij} \geq 0.15$) are displayed, and the individual components are displayed in the format $\{P_{ij}$ - j^{th} URL$\}$. The sessions were assigned to the closest cluster and the session clusters or profiles were examined qualitatively. The results are summarized in Table 5, which also lists the cardinality and the intra-cluster distance for all clusters.

Table 4. Examples of Profiles Discovered by RF$C-$MDE from Maryland Data

i	$\mathbf{P}_i$
2	{.25 - /~mshadl1/Other_Links.html} {.75 - /~mshadl1/profiler.html} {.17 - /~mshadl1/Episode_Guide.html}
4	{.24 - /courses/undergraduate/201/fall98/lectures/index.shtml} {.26 - /courses/undergraduate/201/fall98/projects/index.shtml} {.21 - /courses/undergraduate/201/fall98/projects/p4/index.shtml} {.21 - /courses/undergraduate/201/fall98}
8	{.68 - /~sli2/tetris} {.77 - /~sli2/directory.html} {.67 - /~sli2/tetris/content.html} {.19 - /~sli2/cube/content.html}
9	{.45 - /agents} {0.19 - /agents/news}
12	{.98 - /} {.19 - /people/faculty/faculty.shtml}

The results show that RF$C-$MDE succeeded in delineating many different profiles in the user sessions. Except for the 11^{th} cluster, all clusters correspond to real profiles reflecting distinct user interests. The profiles seem to reflect the intersts of leisurly users that browse the world wide Web in the late evening, as most profiles reflect an interest in free internet games offered by some of the department's graduate students (Profiles No. 2, 5, 6, and 8). Also profile No. 10 reflects an interest in a large set of pictures of a famous supermodel, posted on a graduate student's homepage. Note that many sessions that do not belong to any profile are lumped in the 11^{th} profile which is easily recognized as a spurious cluster by using the quantitative evaluation measures. In fact, this particular cluster had no significant URLs

Table 5. Summary of RF$C-$MDE User Session Profiles for Maryland Data

i	$\lvert\mathcal{X}_i\rvert$	description	$\bar{D}_{Wi}$
1	13	Graduate course and degree enquieries (/www/courses/graduate) and (/www/graduate)	0.78
2	48	Accesses to ~mshadl1 pages (graduate student's web page about the Hit TV series "Profiler")	0.49
3	20	Accesses to ~sletsc1 pages (a page about model ceramic cottages)	0.64
4	58	inquiries about undergraduate course No. 201	0.81
5	27	Accesses to ~sli2 pages, particularly (/~sli2/cube) (a free game offered by a graduate student)	0.43
6	20	Accesses to (/ etoton1/games) (a free game offered by another graduate student)	0.27
7	15	Accesses to / ebert (professor) pages	0.68
8	79	Accesses to ~sli2 pages, particularly (/~sli2/tetris) (a free game offered by a graduate student)	0.55
9	58	inquiries about agents (/agents) (The Intelligent Software Agents page)	0.71
10	19	Accesses to (/ rmobar/Claudia.html) (Supermodel pictures offered by a graduate student)	0.35
11	200	Mixture of unrelated accesses that don't make a strong profile	0.99
12	42	Short sessions mostly limited to main page and faculty list	0.66

($P_{ij} < 0.15$ for all j) and its intra-cluster distance (0.99) was even higher than the total average pairwise distance of all sessions (0.98). In addition to the intra-cluster distance, the robust weights, w_{ij}, are extremely useful for extracting the core members of each class, as well as for distinguishing between strong and spurious profiles. In fact, only the core members of each cluster are expected to have a significant robust weight. For example when only sessions with weights exceeding 0.029 are considered, only profiles Nos. 5, 6, 8, 9, 10, and 12 still have more than 10 members, hence making a strong profile. Note that the weak profile No. 11 was eliminated as all its members have low robust weight. Also the following session from profile No. 5, having low robust weight, was considered as noise, obviously, because it is not clear that the user's interest relates to that of the 5^{th} profile:

{/~sli2/cube/cube.html, /~evans/Hemingway1.html,
/~cef/science96/drift.html}

The Non Euclidean Relational Fuzzy $C-$Means (NERF) [16] was also used to cluster the sessions relation matrix, and resulted in only 4 significant profiles, as shown in Table 6. RF$C-$MDE fared better than NERF in the sense that the spurious cluster (No. 3) found by NERF, which corresponds to the 11^{th} profile found by RF$C-$MDE, had 264 more sessions. Also, NERF completely missed the clusters corresponding to the 3^{rd}, 4^{th}, 7^{th}, 8^{th}, 9^{th}, and

10^{th} profiles, and lumped the 2^{nd} and 6^{th} profiles found by RF$C-$MDE into a single cluster (This corresponds to the 3^{rd} profile found by NERF). Moreover, as can be inferred from the higher cardinality and the average intra-cluster distance for comparable profiles (such as the 5^{th} and 1^{st} profiles found by RF$C-$MDE and NERF respectively), NERF's clusters tend to contain more irrelevent sessions or noise.

As in the previous experiment with the University of Missouri data, We note that as a by-product of the clustering process, associations between different URL addresses on the UMBC site can easily be inferred from the resulting robust profiles. In general, the URLs that are present in the same profile tend to be visited together in the same session.

Table 6. Summary of NERF User Session Profiles for Maryland Data

i	$\lvert\mathcal{X}_i\rvert$	description	$\overline{D}_{Wi}$
1	13	Graduate enquieries (/www/courses/graduate) and (/www/graduate)	0.78
2	112	Accesses to ~sli2 pages, particularly (/~sli2/cube)	0.62
3	66	Accesses to (~mshadl1) and (/ etoton1/games)	0.67
4	464	Mixture of unrelated accesses that don't make a strong profile	0.99
5	35	Short sessions mostly limited to main page	0.58

4 The Relational Fuzzy c Medoids Algorithm (FCMdd)

Let $X = \{\mathbf{x}_i | i = 1, 2, \ldots, n\}$ be a set of n objects. Each object may or may not be represented by a feature vector. Let $r(\mathbf{x}_i, \mathbf{x}_j)$ denote the dissimilarity between object $\mathbf{x}_i$ and object $\mathbf{x}_j$. Let $\mathbf{V} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_c\}, \mathbf{v}_i \in X$ represent a subset of X with cardinality c, i.e., $\mathbf{V}$ is a c-subset of X. Let X^c represent the set of all c-subsets $\mathbf{V}$ of X. The Fuzzy c Medoids Algorithm (FCMdd) minimizes:

$$J_m(\mathbf{V}; X) = \sum_{j=1}^{n} \sum_{i=1}^{c} u_{ij}^m \; r(\mathbf{x}_j, \mathbf{v}_i), \tag{12}$$

where the minimization is performed over all $\mathbf{V}$ in X^c. In (12), u_{ij} represents the fuzzy [5], or possibilistic [15, 23] membership of $\mathbf{x}_j$ in cluster i. The membership u_{ij} can be defined heuristically in many different ways. For example, we can use the FCM [5] membership model given by:

$$u_{ij} = \frac{\left(\frac{1}{r(\mathbf{x}_j, \mathbf{v}_i)}\right)^{1/(m-1)}}{\sum_{k=1}^{c} \left(\frac{1}{r(\mathbf{x}_j, \mathbf{v}_k)}\right)^{1/(m-1)}}, \tag{13}$$

where $m \in [1, \infty)$ is the "fuzzifier". Since u_{ij} is a function of the dissimilarities $r(\mathbf{x}_j, \mathbf{v}_k)$, it can be eliminated from (12). This is the reason J_m is shown as a function of $\mathbf{V}$ alone. When (12) is minimized, the $\mathbf{V}$ corresponding to the solution generates a fuzzy or possibilistic partition via an equation such as (13). However, (12) cannot be minimized via the alternating optimization technique, because the necessary conditions cannot be derived by differentiating it with respect to the medoids. (Note that the solution space is discrete.) Thus, strictly speaking, an exhaustive search over X^c needs to be used. However, following Fu's [11] heuristic algorithm for a crisp version of (12), we describe the following fuzzy algorithm (FCMdd) that minimizes (12).

The Fuzzy c-Medoids Algorithm (FCMdd)

Fix the number of clusters c; Set $iter = 0$;
Pick initial medoids $\mathbf{V} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_c\}$ from X^c;
Repeat
  Compute memberships u_{ij} for $i = 1, 2, \ldots, c$ and
    $j = 1, 2, \ldots, n$, by using (13); (A)
  Store the current medoids: $\mathbf{V}^{old} = \mathbf{V}$;
  Compute the new medoids $\mathbf{v}_i$ for $i = 1, 2, \ldots, c$:
    $q = \underset{1 \le k \le n}{\operatorname{argmin}} \sum_{j=1}^{n} u_{ij}^m \, r(\mathbf{x}_k, \mathbf{x}_j)$; $\mathbf{v}_i = \mathbf{x}_q$; (B)
  $iter = iter + 1$;
Until $(\mathbf{V}^{old} = \mathbf{V}$ or $iter = MAX\_ITER)$.

The crisp version of FCMdd above, which we call the Hard c Medoids (HcMdd) algorithm, can be obtained by replacing step (A) with:

$$q = \underset{1 \le k \le c}{\operatorname{argmin}} \, r(\mathbf{x}_j, \mathbf{v}_k); \quad u_{ij} = \begin{cases} 1 & \text{if } i = q \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

The above algorithm falls in the category of Alternating Cluster Estimation [38] paradigm, and is not *guaranteed* to find the global minimum. It is advisable to try many random initializations to increase the reliability of the results. We have experimented with three different ways of initializing the medoids. The first way is to pick all the medoid candidates randomly. We call this method Initialization I. The second way is to pick the first candidate as the object that is most central to the data set, and then pick each successive one by one in such a way that each one is most dissimilar to all the medoids that have already been picked. This makes the initial medoids evenly distributed. We refer to this procedure as Initialization II.

Initialization II for FCMdd

Fix the number of medoids $c > 1$;
Compute the first medoid:
$q = \underset{1 \le j \le n}{\operatorname{argmin}} \sum_{i=1}^{n} r(\mathbf{x}_j, \mathbf{x}_i)$; $\quad \mathbf{v}_1 = \mathbf{x}_q$;
Set $V = \{\mathbf{v}_1\}$, $iter = 1$;
Repeat
$iter = iter + 1$;
$q = \underset{1 \le i \le n; \mathbf{x}_i \notin V}{\operatorname{argmax}} \min_{1 \le k \le |V|} r(\mathbf{v}_k, \mathbf{x}_i)$; $\quad \mathbf{v}_{iter} = \mathbf{x}_q$;
$V = V \cup \{\mathbf{v}_{iter}\}$;
Until $(iter = c)$.

For a given data set, the initialization produced by Initialization II is always fixed. Sometimes a bit of randomness might be desirable. In the third initialization strategy, we add randomness by picking the first medoid candidate randomly. The rest of the medoids are selected the same way as in Initialization II. We call this method Initialization III. The computational complexity of Initialization I, II and III are $\mathcal{O}(c)$, $\mathcal{O}(nc^2)$ and $\mathcal{O}(nc^2)$ respectively. We found that both Initialization II and III work well in practice.

The fuzzifier m in FCMdd determines the degree of fuzziness of the resulting clusters. Since the medoid always has a membership of 1 in the cluster, raising its membership to the power m has no effect. Thus, when m is high, the mobility of the medoids from iteration to iteration may be lost, because all memberships become very small except the one corresponding to the current medoid. For this reason, we recommend a value between 1 and 1.5 for m.

It can be seen from step (B) of FCMdd that the complexity of the algorithm is $\mathcal{O}(n^2)$, where n is the number of input objects. However, this is too expensive for most Web mining applications. To overcome this problem, we can modify step (B) of FCMdd so that it examines only a subset of objects while updating the medoid for cluster i. The subset we choose is the set of p objects in X that correspond to the top p highest membership values in cluster i. We denote this subset by $X_{(p)i}$. The subsets $X_{(p)i}$, $i = 1, 2, \ldots, c$, can be identified during the membership updating step, i.e., in step (A) of the algorithm. This increases the complexity of step (A) to $\mathcal{O}(ncp)$. However, the complexity of step (B) is reduced to $\mathcal{O}(ncp)$. Therefore, the overall complexity is linear in the number of objects. The value of p should be proportional to the dimensionality d of the data, e.g. $2d$. However, when the data is relational, d has no meaning. In such cases, p could be chosen to be much smaller than the average number (n/c) of points in a cluster. The modified FCMdd algorithm is summarized below.

Linearized Fuzzy c-Medoids Algorithm (LFCMdd)

Fix the number of clusters c; Set iter = 0;
Pick the initial set of medoids $\mathbf{V} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_c\}$ from X^c;
Repeat
  Compute memberships u_{ij} for $i = 1, 2, \ldots, c$, and $j = 1, 2, \ldots, n$,
    by using (13) and identify $X_{(p)i}$, $i = 1, 2, \ldots, c$. (A)
  Store the current medoids: $\mathbf{V}^{old} = \mathbf{V}$;
  Compute the new medoids $\mathbf{v}_i$ for $i = 1, 2, \ldots, c$:
    $q = \underset{\mathbf{x}_k \in X_{(p)i}}{\operatorname{argmin}} \sum_{j=1}^{n} u_{ij}^m \; r(\mathbf{x}_k, \mathbf{x}_j)$ $\quad$ $\mathbf{v}_i = \mathbf{x}_q$; (B)
  $iter = iter + 1$;
Until $(\mathbf{V}^{old} = \mathbf{V}$ or $iter = MAX\_ITER)$.

4.1 Robust Versions of FCMdd

It is well-known that algorithms that minimize a Least-Squares type objective function are not robust [7, 37]. In other words, a single outlier object could lead to a very unintuitive clustering result. To overcome this problem, we design an objective function for a robust version of FCMdd based on the Least Trimmed Squares idea [22, 37], we use the membership function in (13). Substituting the expression for u_{ij} in (13) into (12), we obtain:

$$J_m(\mathbf{V}; \mathbf{X}) = \sum_{j=1}^{n} \left(\sum_{i=1}^{c} (r(\mathbf{x}_j, \mathbf{v}_i))^{1/(1-m)} \right)^{1-m} = \sum_{j=1}^{n} h_j, \tag{15}$$

where

$$h_j = \left(\sum_{i=1}^{c} (r(\mathbf{x}_j, \mathbf{v}_i))^{1/(1-m)} \right)^{1-m} \tag{16}$$

is $1/c$ times the harmonic mean of the dissimilarities $\{r(\mathbf{x}_j, \mathbf{v}_i)) : i = 1, \ldots, c\}$ when $m = 2$. The objective/criterion function of the Robust Fuzzy c Medoids (RFCMdd) algorithm is obtained by modifying (15) as follows:

$$J_m^T(\mathbf{V}; \mathbf{X}) = \sum_{k=1}^{s} h_{k:n}. \tag{17}$$

In (17), $h_{k:n}$ represents the k-th item when $h_j, j = 1, 2, \ldots, n$, are arranged in ascending order, and $s < n$. The value of s is chosen depending on how many objects we would like to disregard in the clustering process. This allows the clustering algorithm to ignore outlier objects while minimizing the objective function. For example, when $s = n/2$, 50% of the objects are not considered in the clustering process, and the objective function is minimized when we pick c medoids in such a way that the sum of the harmonic-mean dissimilarities of 50% of the objects is as small as possible. We can design the following heuristic algorithm to minimize (17).

The Robust Fuzzy c Medoids Algorithm (RFCMdd)

Fix the number of clusters c, and the fuzzifier m;
Pick the initial medoids $\mathbf{V} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_c\}$ from X^c;
$iter = 0$;
Repeat
  Compute harmonic dissimilarities h_j for $j = 1, 2, \ldots, n$, using (16);
  Sort h_j, $j = 1, 2, \ldots, n$ to create $h_{j:n}$;
  Keep the s objects $\mathbf{x}_{1:n}, \ldots, \mathbf{x}_{s:n}$, corresponding to the first s $h_{j:n}$;
  Compute memberships $u_{ij:n}$ for $i = 1, 2, \ldots, c$ and $j : n = 1, 2, \ldots, s$,
    by using (13), and identify $X_{(p)i}$, $i = 1, 2, \ldots, c$;
  Store the current medoids: $\mathbf{V}^{old} = \mathbf{V}$;
  Compute the new medoids $\mathbf{v}_i$ for $i = 1, 2, \ldots, c$:
    $q = \underset{\mathbf{x}_{k:n} \in X_{(p)i}}{\operatorname{argmin}} \sum_{j=1}^{s} u_{ij:n}^m \, r(\mathbf{x}_{k:n}, \mathbf{x}_{j:n})$; $\mathbf{v}_i = \mathbf{x}_q$;
  $iter = iter + 1$;
Until $\left(\mathbf{V}^{old} = \mathbf{V}$ or $iter = MAX\_ITER\right)$.

As mentioned above, the choice of the retention ratio, s/n, should reflect the percentage of noise in the data. If the noise proportion is higher than 50%, we cannot guarantee that we will obtain the correct estimates for the parameters such as cluster center, variance, etc [7, 37]. A common approach in robust statistics is to assume that the noise proportion is 50% and then apply a correction to the estimate *after* the parameters have been (robustly) estimated [37]. For a known distribution (such as a Gaussian), the correction is given by a simple formula. In our application, since the data is relational, such corrections can be difficult to apply, except in a heuristic manner. Another option is to estimate the retention ratio when it is not known in advance. When estimation is not possible, our recommendation is that we estimate or pick a lowerbound for retention ratio and use it. This will at least ensure that the estimates are not affected by outliers. It is also possible to optimize the objective function with respect to the retension ratio. See for example [28, 30].

Interestingly, the worst-case complexity of the RFCMdd algorithm still remains $\mathcal{O}(n\log n)$. This is a good result, considering that robust algorithms are very expensive. A robust fuzzy clustering algorithm based on the Least Median of Squares idea, presented in [27], can easily me extended to relational clustering as well. In this case we simply replace the summation in (17) by the median. In other words, we use

$$J_m^M(\mathbf{V}; \mathbf{X}) = \underset{1 \le k \le n}{\text{median}} \; h_{k:n}. \tag{18}$$

A genetic algorithm was used to optimize this criterion in [27].

4.2 Experimental Results

To validate the capacity of our algorithms to extract user access patterns, we ran several experiments comparing FCMdd with NERF . Both the naive and the linearized implementations of FCMdd were used. The experiments were done on a number of Web server logs obtained from servers at the University of Maryland - Baltimore County (UMBC). We report here a subset of our experimental results, on log sizes that represent a quarter days activity to five days of activity in April 2000, on the UMBC CSEE Web server. Instead of reporting the number of hits in each log, we report here the number of sessions generated from a log since the clustering algorithm operates on the sessions. In the case of FCMdd, we overspecify the number of clusters to be 50. The same number of clusters (50) was used for NERF as well.

The intracluster distance for most clusters (average 0.343) was much smaller than the intercluster distance (average, 0.996), although there are a few exceptions, such as cluster 0 (0.998). This cluster represented an aggregation of all course related URLs.

Given that in the linearized implementation, we are using only a fraction of the possible medoid candidates at each update step, one might expect that the clusters formed may not be as good as the regular implementation. However, we found that the LFCMdd generated better clusters than the regular implementation for many of our logs. In fact, the average intra cluster distance was 0.343 for the linearized version as opposed to 0.435 for the regular implementation. We speculate that this is because of the "local" nature of the search used by the linearized version. The linearized version confines its search for the medoid of a cluster to the neighborhood of the current medoid. This means that given a good initialization, this method can find good "localized" clusters. The regular implementation does a more global search and sometimes could get stuck in strange minima.

We provide brief descriptions of the clusters generated by our algorithm. Table 7 lists some of the clusters found. Recall that the clusters were generated by overspecifying the number of clusters to be 50 and then determining the actual number c_{actual} using the procedure described in Section 2.4. When the cardinality of a cluster is too low (in our case when less than 30), the cluster was considered as not having enough support, and thus was not listed. For illustration purposes, we show here the top level URLs found, and in the table show the sum of the strengths of the underlying URLs. For instance, if a cluster has the URL ∼plusquel/ with strength 0.8 and ∼plusquel/cmsc310 with strength 0.7, we will report it here as ∼plusquel with strength 1.5.

The clusters seemed fairly consistent across logs worth several days. In other words, similar (though not identical) clusters were formed as we mined different sized logs from the same time period. In our analysis, we chose to disregard clusters that had a very small cardinality (number of sessions in them). In traditional data mining jargon, one would say that there was not

enough support for such clusters. We also ignored those clusters that did not have a strong URL profile as explained in Section 2.4.

- Clusters 0 (/agentslist/) and 20 (/agents/) contain user sessions that accessed the pages maintained by the Agents group at UMBC.
- Cluster 2 (/help/oracle/) represents user sessions that accessed the help pages of oracle8, likely to be from students enrolled in the undergraduate and graduate database courses.
- Cluster 6 (/www/) represents user sessions that accessed an older version of the CSEE Web pages.
- Cluster 9 (/471/) represents users who want to access the CMSC 471 course pages. It contains hits to pages containing current information, lectures and notes.
- Clusters 11 (/cgi-bin/) represents access to the queries provided on the server using CGI scripts.
- Cluster 19 (/∼ugrad) represents users who accessed Web pages that provides brochures and admission information for undergraduates.
- Clusters 21 (/courses/) contains user sessions that access information about the courses offered by the CS department. Given the enrollment differences, a larger support is found for /courses/undergrad/ as opposed to /courses/graduate/, as expected.
- Clusters 3 (/∼kalpakis), 13 (/∼qlu2) and 18 (/∼mikeg) correspond to user sessions that accessed home pages of individual users.
- The remaining lusters have cardinalities that are too small to be included in the study.

5 Creating Personalized Web Pages Using Adaptive Web Servers

Traditionally, adaptive Web site development has been limited to the use of data obtained from the user by filling out "registration" type forms. Sites such as http://my.netscape.com/ and http://my.cnn.com/ create a personalized Web page, based only on such declarative information. Typically, these sites list all possible categories of information and leave it to the user to select or de-select a particular category of information. The user can also decide the layout by placing items of relative importance above other items. Intelligence derived from tracking a single user can be used to create adaptive Web pages that show his/her interests as well as recommend similar pages. Amazon.com uses collaborative filtering to direct customers to items bought by other customers who purchased the same item. These methods either use declarative information or track a single user to obtain profiles. These methods have raised privacy concerns since they often deal with personal and identifying information.

Table 7. CSEE Logs Analysis using Linear FCMdd Algorithm (clusters with more than 30 sessions)

Cluster	Cardinal	URLs	URLs	Deg
0 - /agentslist/	628	3864	{/agentslist/*}	1.511
			{/agentslist/archive/*}	1.449
2 - /help/oracle8/	41	241	{/help/*}	6.976
			{/help/oracle8/*}	6.585
			{/help/oracle8/server803*}	4.488
			{/help/oracle8/server803/ A54654_01/*}	1.098
3 - /~kalpakis/	46	137	{/~kalpakis/*}	3.543
			{/~kalpakis/Courses/*}	1.761
			{/~kalpakis/Courses/441/*}	1.326
6 -/www/	32	104	{/www/*}	3.156
			{/www/graduate/*}	2.531
			{/www/graduate/rpg/*}	1.156
9 -/471/	60	893	{/471/*}	14.167
			{/471/current/*}	5.45
			{/471/current/lectures/*}	4.333
			{/471/lectures/*}	5.25
			{/471/lectures/uninformed-search/*}	1.05
			{/471/notes/*}	3.333
			{/471/notes/7/*}	1.05
11 -/cgi-bin/	76	358	{/cgi-bin/*}	4.171
			{/cgi-bin/raw?url=http:/*}	3.776
			{/agents/*}	2.697
13 -/~qlu2/	49	12	{/~qlu2/*}	1.837
16 -/~sli2/	236	203	{/~sli2/*}	13.86
			{/~sli2/cube/*}	9.932
			{/~sli2/plot*}	2.085
18 -/~mikeg/	81	90	{/~mikeg/*}	3.012
20 -/agents/	466	1544	{/agents/*}	3.406
21 -/courses/	1048	3443	{/courses/*}	7.889
			{/courses/undergraduate/*}	6.776
			{/courses/undergraduate/201/*}	2.512

Our work uses a very simple way to profile users without revealing any personal information about them. In our system, an apache module places a cookie on the users machine. This cookie is generated using the *mod_usertrack* option which generates a random unique number for a user and places it as the cookie on the users machine. The logs generated also do not contain user ids as *identd* is not used when a hit is registered. Hence, creating a mapping between user ids and their cookie information is also not possible. A user visiting a Web page, is viewed as one of the users belonging to a cluster which has behavior patterns similar to his or her own. This method has the

advantage of not revealing any private profile information about a user, but at the same time generating a *personalized* page for the user.

Assume that the user is interested in the class pages for CMSC 771 and CMSC 661. This information is inferred from her inclusion in the groups of users frequenting the pages for CMSC 771 and CMSC 661. Thus the list of URLs, she might be interested in is compiled on the basis of the groups, she belongs to. Page titles and snippets for the URL are also displayed along with the link to the page to make her task of identifying the page simpler. Thus, a user coming to the UMBC CS page would be shown a lists of URLs which she frequents on the UMBC CS server. A link to the default CS page is also provided, in case the user is interested in looking up some information not presented on his personal page.

Recent experiments in comparing the usage of IP addresses verses cookies have concluded cookies to generate more accurate user sessions than IP addresses.

Cookies are unique to a user and are placed on the user's machine, the first time the user accesses the particular Web server. Every access to the Web server after the first hit, registers the hit belonging to the user's cookie. No mapping between a user and a cookie is maintained, hence keeping the users identification unknown.

5.1 Experimental Setup of Personalizing Tool and Results

The module is coded in Mod Perl and runs on Apache HTTP Server 1.2.5. Cookies with a unique number were generated using the *mod_usertrack* module and placed on the user's machine. Every hit on the website registers an entry on the server logs along with the cookie information.

Session files consisting of the unique session numbers and the cookie related to the particular session were generated. Clusters are generated for the Web logs using the sessions. A cluster file consists of the mapping of the session number and the cluster it belongs to. Having assigned sessions to a cluster, profiles have to be generated for each cluster. A profile for a cluster is defined as the most common URLs belonging to sessions, which in turn belong to the cluster. An apache module generates a Web page on the fly for a user coming to the site. The user's cookie is examined to determine the sessions corresponding to his or her traversal patterns. If no cookie is present or if no matching sessions for the cookie are found, the user is shown the default page. Otherwise, further processing is carried out to generate their personalized page.

For each session matching the cookie, corresponding clusters are obtained. A user might have multiple sessions recorded in the logs. Different sessions could point to different clusters i.e. traversal patterns or may point to the same cluster. In case of different traversal patterns, URLs from different clusters are displayed on the personalized page, grouped by their appearance in

the clusters. A snippet of information is also provided along with the link to the page.

Clusters for the experiment were generated from logs obtained from servers at the Computer Science Department at UMBC. In the snapshot provided, the user falls into multiple categories. The user accesses the pages related to Knowledge Query and Manipulation Language (KQML), the pages related to the CIKM 2000 conference and also the pages related to the 471 class at UMBC. These different categories denote the different *modes* in which the user uses the CS Web pages. A user might show a single traversal pattern on a Web server or a pattern which is a combination of different traversal patterns. Figure 1 shows links related to the user's traversal pattern on the CS server.

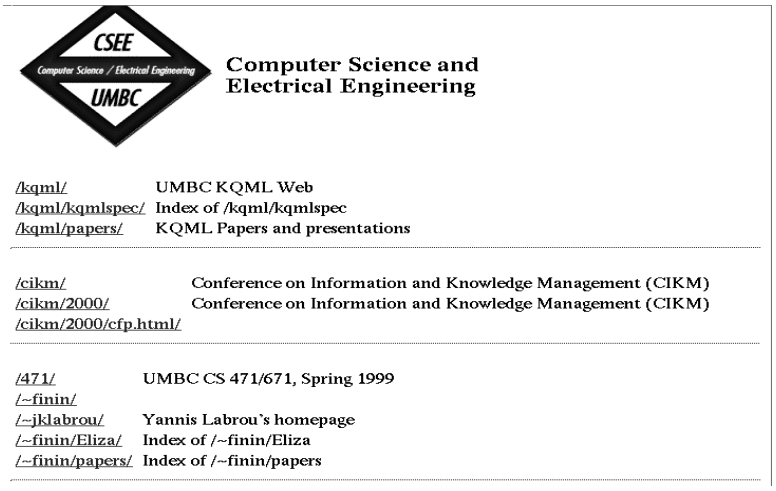

Fig. 1. Adaptive Web page for a Web user

6 Conclusions

We have presented a new approach for the automatic discovery of user session profiles in Web log data based on robust fuzzy relational clustering. We defined the notion of a "user session" as being a temporally compact sequence of Web accesses by a user. A new similarity measure to analyze session profiles is presented which captures both the individual URLs in a profile as well as the structure of the site. we have presented new relational fuzzy clustering algorithms (RFCMdd and RF$C-$MDE) that have the capacity to handle complex noisy and fuzzy Web usage data, and used them to successfully cluster the sessions extracted from real server access logs into typical user session profiles, and even to identify the noisy sessions and profiles. The resulting

clusters are evaluated subjectively, as well as based on standard statistical criteria. The criteria used were the intra-cluster and inter-cluster distances, and the significance of the components of a session "profile" vector which also summarizes the typical sessions in each cluster. Our experiments on real data sets showed that our approach outperformed NERF in extracting profiles from the Web access logs, and in identifying noisy sessions.

We also presented an approach to personalize the Web space by generating pages dynamically based on off-line clustering of Web logs. Web logs were sessionized using cookies. These adaptive Web pages can reduce network traffic by directly showing a user to the page of his or her interest without them having to look around for it. Such an approach also mitigates the issue of security since no personal or declarative information is obtained from a user to present "personalized" pages. In ongoing work, we are examining the scalability of the system and extending it to examine logs and add results in an incremental fashion to existing results. Also, a study as to whether the users find such personalized pages to be relevant and to what extent, is being carried out.

Note that in some applications, the frequency of accesses to a Web page (or alternately, the viewing time) within the same session may be important. In that case, the definition of $\mathbf{s}_j^{(k)}$ should be modified to the number of times the j^{th} URL is accessed in the k^{th} session (or alternately, some measure of relative viewing time). In ongoing experiments, we are also looking into a multi-resolution profiling approach where clustering is applied recursively on the profiles found in previous runs.

Acknowledgements

Partial support of this work by the National Science Foundation Grants IIS 9800899 and IIS 9801711 is gratefully acknowledged. This work was also partially supported by a Faculty Research Initiation Grant from the University of Memphis to Olfa Nasraoui.

References

1. R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proceedings of the 20th VLDB Conference*, pages 487–499, Santiago, Chile, 1994.
2. R. Armstrong, T. Joachims D. Freitag, and T. Mitchell. Webwatcher: A learning apprentice for the World Wide Web. In *Proceedings of the AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments*, pages 6–13, Stanford, CA, March 1995.
3. G. Arocena and A. Mendelz. Weboql: Restructuring documents, databases, and web. In *Proc. IEEE Intl. Conf. Data Engineering '98*. IEEE Press, 1998.
4. P. Bajcsy and N. Ahuja. Location- and density-based hierarchical clustering using similarity analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:1011–1015, 1998.

5. J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms.* Plenum Press, New York, 1981.
6. R. Cooley, B. Mobasher, and J. Srivasta. Web Mining: Information and pattern discovery on the World Wide Web. In *Proc. IEEE Intl. Conf. Tools with AI*, pages 558–567, Newport Beach, CA, 1997.
7. R. N. Davé and R. Krishnapuram. Robust clustering methods: A unified view. *IEEE Transactions on Fuzzy Systems*, 5(2):270–293, 1997.
8. E. Diday. La methode des nuees dynamiques. *Rev. Stat. Appliquee*, XIX(2):19–34, 1975.
9. U. Fayad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery: An overview. In U. Fayad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining.* AAAI/MIT Press, 1996.
10. H. Frigui and R. Krishnapuram. Clustering by competitive agglomeration. *Pattern Recognition*, 30(7):1223–1232, 1997.
11. K. S. Fu. *Syntactic Pattern Recognition and Applications.* Academic Press, San Diego, CA, 1982.
12. K. C. Gowda and E. Diday. Symbolic clustering using a new similarity measure. *IEEE Transactions on Systems, Man, and Cybernetics*, 20:368–377, 1992.
13. S. Guha, R. Rastogi, and K. Shim. CURE: An efficient algorithm for large databases. In *Proceedings of SIGMOD '98*, pages 73–84, Seattle, June 1998.
14. D. E. Gustafson and W. C. Kessel. Fuzzy clustering with the fuzzy covariance matrix. In *Proccedings of IEEE CDC*, pages 761–766, San Diego, California, 1979.
15. R. J. Hathaway and J. C. Bezdek. Switching regression models and fuzzy clustering. *IEEE Transactions on Fuzzy Systems*, 1(3):195–204, 1993.
16. R. J. Hathaway and J. C. Bezdek. NERF c-means: Non-Euclidean relational fuzzy clustering. *Pattern Recognition*, 27:429–437, 1994.
17. R.J. Hathaway, J.W. Devenport, and J.C. Bezdek. Relational dual of the c-means clustering algorithms. *Pattern Recognition*, 22(2):205–212, 1989.
18. A. Joshi, C. Punyapu, and P. Karnam. Personalization and asynchronicity to support mobile web access. In *Proc. Workshop on Web Information and Data Management, 7^{th} Intl. Conf. on Information and Knowledge Management*, November 1998.
19. A. Joshi, S. Weerawarana, and E. Houstis. On disconnected browsing of distributed information. In *Proceedings of IEEE Intl. Workshop on Research Issues in Data Engineering (RIDE)*, pages 101–108, Birmingham, UK, 1997.
20. L. Kaufman and P. J. Rousseeuw. Clustering by means of medoids. In Y. Dodge, editor, *Statistical Data Analysis Based on the L_1 Norm*, pages 405–416. North Holland/Elsevier, Amsterdam, 1987.
21. L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data, An Itroduction to Cluster Analysis.* John Wiley & Sons, Brussels, Belgium, 1990.
22. J. Kim, R. Krishnapuram, and R. N. Davé. Application of the least trimmed squares technique to prototype-based clustering. *Pattern Recognition Letters*, 17:633–641, 1996.
23. R. Krishnapuram and J. M. Keller. A possibilistic approach to clustering. *IEEE Transactions on Fuzzy Systems*, 1(2):98–110, 1993.
24. O. Nasraoui, H. Frigui, R. Krishnapuram, and A. Joshi. Mining web access logs using relational competitive fuzzy clustering. In *Eighth International Fuzzy Systems Association Congress*, Hsinchu, Taiwan, Aug. 1999.

25. O. Nasraoui and R. Krishnapuram. Crisp interpretation of fuzzy and possibilistic clustering algorithms. In *3rd European Congress on Intelligent Techniques and Soft Computing*, volume 3, pages 1312–1318, Aachen, Germany, Aug. 1995.
26. O. Nasraoui and R. Krishnapuram. A robust estimator based on density and scale optimization, and its application to clustering. In *IEEE International Conference on Fuzzy Systems*, pages 1031–1035, New Orleans, LA, Sep. 1996.
27. O. Nasraoui and R. Krishnapuram. A genetic algorithm for robust clustering based on a fuzzy least median of squares criterion. In *Proceedings of NAFIPS'97*, pages 217–221, Syracuse, NY, Sept. 1997.
28. O. Nasraoui and R. Krishnapuram. Mining web access logs using a relational clustering algorithm based on a robust estimator. In *Proc. of the Eighth International World Wide Web Conference*, pages 40–41, Toronto, 1999.
29. O. Nasraoui, R. Krishnapuram, H. Frigui, and Joshi A. Extracting web user profiles using relational competitive fuzzy clustering. *International Journal on Artificial Intelligence Tools*, 9(4):509–526, 2000.
30. O. Nasraoui, R. Krishnapuram, and A. Joshi. Relational clustering based on a new robust estimator with application to web mining. In *Proceedings of the North American Fuzzy Information Society*, pages 705–709, New York City, 1999.
31. R. T. Ng and J. Han. Efficient and effective clustering methods for spatial data mining. In *Proceedings of the 20th VLDB Conference*, pages 144–155, Santiago, Chile, Sept. 1994.
32. O.Zaiane and J. Han. Webml: Querying the world-wide web for resources and knowledge. In *Proc. Workshop on Web Information and Data Management, 7^{th} Intl. Conf. on Information and Knowledge Management*, 1998.
33. M. Perkowitz and O. Etzioni. Adaptive web sites: an ai challenge. In *Proc. Intl. Joint Conf. on AI – IJCAI97*, 1997.
34. M. Perkowitz and O. Etzioni. Adaptive web sites: Automatically synthesizing web pages. In *Proc. AAAI 98*, 1998.
35. G. D. Ramkumar and A. Swami. Clustering data without distance functions. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 21:9–14, 1998.
36. M. Roubens. Pattern classification problems and fuzzy sets. *Fuzzy Sets and Systems*, 1:239–253, 1978.
37. P. J. Rousseeuw and A. M. Leroy. *Robust Regression and Outlier Detection*. John Wiley and Sons, New York, 1987.
38. T. A. Runkler and J. C. Bezdek. ACE: A tool for clustering and rule extraction. *IEEE Transactions on Fuzzy Systems*, 1999.
39. E. H. Ruspini. Numerical methods for fuzzy clustering. *Information Science*, 2:319–350, 1970.
40. C. Shahabi, A. M. Zarkesh, J. Abidi, and V. Shah. Knowledge discovery from user's web-page navigation. In *Proceedings of the IEEE Intl. Workshop on Research Issues in Data Engineering (RIDE)*, pages 20–29, Birmingham, UK, 1997.
41. U. Shardanand and P. Maes. Social information filetering: Algorithms for automating 'word of mouth'. In *Proc. CHI'95 Conference on Human Factors in Computing Systems*, New York, 1995. ACM Press.
42. P. H. A. Sneath and R. R. Sokal. *Numerical Taxonomy - The Principles and Practice of Numerical Classification*. W. H. Freeman, San Francisco, 1973.

43. Y. El Sonbaty and M. A. Ismail. Fuzzy clustering for symbolic data. *IEEE Transactions on Fuzzy Systems*, 6:195–204, 1998.
44. L. Terveen, W. Hill, and B. Amento. PHOAKS - a system for sharing recommendations. *Communications of the ACM*, 40(3):59–62, 1997.
45. M. P. Windham. Numerical classification of proximity data with assignment measures. *Journal of Classification*, 2:157–172, 1985.