Non-crisp Clustering by Fast, Convergent, and Robust Algorithms

Vladimir Estivill-Castro and Jianhua Yang

Department of Computer Science & Software Engineering
The University of Newcastle, Callaghan, NSW 2308, Australia

Abstract. We provide sub-quadratic clustering algorithms for generic dissimilarity. Our algorithms are robust because they use medians rather than means as estimators of location, and the resulting representative of a cluster is actually a data item. We demonstrate mathematically that our algorithms converge. The methods proposed generalize approaches that allow a data item to have a degree of membership in a cluster. Because our algorithm is generic to both, fuzzy membership approaches and probabilistic approaches for partial membership, we simply name it non-crisp clustering. We illustrate our algorithms with categorizing WEB visitation paths. We outperform previous clustering methods since they are all of quadratic time complexity (they essentially require computing the dissimilarity between all pairs of paths).

1 Introduction

In a top-down view to clustering [3], the aim is to partition a large data set of heterogeneous objects into more homogeneous classes. Many clustering methods are built around this partitioning approach [17]. Because we are to partition a set into more homogeneous clusters, we need to assess homogeneity. The degree of homogeneity in a group is a criterion for evaluating that the samples in one cluster are more like one another than like samples in other clusters. This criterion is then made explicit if there is a distance between objects. The type of clustering that we find here attempts to find the partition that optimizes a given homogeneity criterion defined in terms of distances. Thus, this type of clustering is also distance-based, but rather than looking for the most similar pair of objects and grouping them together, which is the bottom-up approach of hierarchical clustering, we seek to find a partition that separates into clusters. Variants of the partitioning problem arise as we see different criteria for defining homogeneity and also different measures of similarity. Then, within one variant of the problem, several algorithms are possible. Consider the following optimization criteria for clustering which attempts to minimize the heterogeneity in each group with respect to the group representative.

$$\text{Minimize } M^a(C) = \sum_{i=1}^{n} d(u_i, \text{REP}[u_i, C])^a, \tag{1}$$

L. De Raedt and A. Siebes (Eds.): PKDD 2001, LNAI 2168, pp. 103–114, 2001.

where $a \geq 1$ is a constant, $C \subset U$ is a set of k representatives, REP$[u_i, C]$ is the most similar representative (in C) to u_i and $d(\cdot,\cdot)$ is a measure of dissimilarity. We underline that $d(\cdot,\cdot)$ does not need to be a distance satisfying the axioms of a metric. In particular, while $d(u_i, u_i) = 0$, we do not require that $d(u_i, u_j) = 0$ implies $u_i = u_j$. Also, we do not require the triangle inequality, however, we do expect symmetry, that is $d(u_i, u_j) = d(u_j, u_i)$. These requirements are satisfied by all similarity measures based on computing the cosine of the angle between attribute-vectors of positive-valued features [16,27,28,32,33]. Note that these similarity functions $sim(\cdot,\cdot)$ have a range in $[0, 1]$ and the corresponding dissimilarity is $d(\cdot,\cdot) = 1 - sim(\cdot,\cdot)$, also in the range [0,1].

Equation (1) is not unfamiliar to statistics nor to the Data Mining community. The case $a = 2$ defines the objective function that the k-MEANS method iteratively approximates. This case is the result of applying an analysis of variance using total sum error squares as the loss function. The case $a = 1$ replaces means by medians and the L_2 loss function by the L_1 loss function (the total absolute error). The case $a = 1$ was brought over from the statistics as medoid-based clustering [21]. However, CLARANS is a randomized interchange hill-climber in order to obtain subquadratic algorithmic complexity. CLARANS can not guarantee local optimality. The best interchange hill-climber [11] is the Teitz and Bart heuristic [31] which requires quadratic time. Only in restricted cases, the time complexity of this type of hill-climber has been reduced to subquadratic time (for example, $D = 2$ and Euclidean distance [9]). Because medians are a more robust estimator of location than means, an algorithm optimizing the case $a = 1$ is more resistant to noise and outliers than an algorithm optimizing the case $a = 2$. However, k-MEANS is heavily used because it is fast [1,24,34], despite its many drawbacks documented in the literature [12].

Clustering algorithms optimizing the family of criteria given by Equation (1) search for a subset $C = \{c_1, \ldots, c_k\}$ of k elements in U. In the case $a = 1$, we use medoids to denote discrete medians; that is, the estimator of location for a cluster shall be a member of the data. Previous to the approach presented here, the classification step for medoids performs a crisp classification. That is, classification computes the representative REP$[u_i, C]$ of each data item u_i and each data item belongs to only the cluster of its representative. It seems that robustness to initialization (as it happens in k-HARMONIC MEANS [34]) comes from a combining relaxation of crisp classification with a boosting technique. It seems that to achieve this, each data item should be able to belong to different clusters with a degree of membership (the degree could be a probability as in EXPECTATION MAXIMIZATION, or a fuzzy membership as in FUZZY-c-MEANS).

Thus, our contribution here is to achieve this degree of membership to different clusters. We will propose algorithms and then show their advantages. We prove mathematically that they converge. We show our algorithms are generic (the degree of membership could be a fuzzy-membership function, a harmonic distribution, or even revert to the case of crisp-membership). We show versions requiring sub-quadratic time by using randomization. We apply our algorithms to a case study where other algorithms can not be used [13,18].

2 Non-crisp Clustering Algorithms

Consider the crisp classification step that computes $\text{REP}[u_i, C]$ for each data item u_i. It requires a simple pass through the data and $O(nk)$ computations of $d(\cdot,\cdot)$. This results in a temporary partition of U into k clusters $U_1, \ldots, U_k$. We can generalize crisp classification by assigning a vector $\boldsymbol{\pi}_i \in [0,1]^k$ to each u_i. We consider that the j-th entry $\boldsymbol{\pi}_{ij}$ denotes the degree by which u_i belongs to the j-th cluster. In parallel with Expectation Maximization and fuzzy-c-Means, we will normalize the values so that $\sum_{j=1}^{k} \boldsymbol{\pi}_{ij} = 1$. Crisp classification means the closest representative has its entry in $\boldsymbol{\pi}$ set to 1 while all other clusters have their entry set to zero. Non-crisp classification will mean more than one cluster has its entry different from zero.

To detail more our approach we need to introduce some notation. Given a vector $\boldsymbol{x} \in \Re^k$, let $\text{SORT}(\boldsymbol{x}) \in \Re^k$ denote the vector of sorted entries from $\boldsymbol{x}$ in non-decreasing order (entries with equal values are arranged arbitrarily). Thus, if $j < j'$ then $\text{SORT}(\boldsymbol{x})_j \leq \text{SORT}(\boldsymbol{x})_{j'}$. We use the notation e_j^{T} to denote the j-th canonical vector $(0, \ldots, 0, 1, 0, \ldots, 0)$ that has only the j-th entry equal to 1 and all other equal to zero. This notation allows us to rewrite the loss function in Equation (1) because the minimum operator in $\text{REP}[u_i, C]$ can be replaced by $e_1^{\mathrm{T}} \cdot \text{SORT}(\cdot)$ as follows $M^a(C) = \sum_{i=1}^{n} e_1^{\mathrm{T}} \cdot \text{SORT}(d(u_i, c_1), d(u_i, c_2), \ldots, d(u_i, c_k))$. Moreover, we can already describe a Harmonic set of weights as the degrees of membership. Let $K = \sum_{j=1}^{k} 1/j$, the Harmonic loss function is then $M^H(C) = \frac{1}{K} \sum_{i=1}^{n} (1, 1/2, \ldots, 1/j, \ldots, 1/k)^{\mathrm{T}} \cdot \text{SORT}(d(x_i, c_1), d(u_i, c_2), \ldots, d(u_i, c_k))$. Moreover, we propose here an even more general approach, and let $\omega \in \Re^k$ be any vector with non-negative entries and entries in descending order; then, the non-crisp classification loss function with respect to ω is

$$M^{\omega}(C) = \frac{1}{\|\omega\|_1} \sum_{i=1}^{n} \omega^{\mathrm{T}} \cdot \text{SORT}(d(u_i, c_1), d(u_i, c_2), \ldots, d(u_i, c_k)). \tag{2}$$

(where we denote the 1-norm of a vector $\boldsymbol{x}$ as $\|\boldsymbol{x}\|_1$ and it equals $\sum_{j=1}^{k} |\boldsymbol{x}_k|$). Note that the first canonical vector e_1 and the Harmonic vector with $1/j$ in its j-th entry are special cases for ω. The vector $\boldsymbol{\pi}_i$ giving the degree of membership of u_i is defined by $\pi_{ij} = d(u_i, c_j)\omega_j / \|\omega\|_1$.

Recently, by using randomization several variants of clustering algorithms for optimizing instances of the criteria in Equation (1) with $a = 1$ have emerged [10]. These variants are subquadratic robust clustering algorithms. These randomized algorithms have been shown mathematically and empirically to provide robust estimators of location [10]. This is because randomization is not sampling. Sampling reduces the CPU-requirements by using a very small part of the data, and the accuracy suffers directly with the size of the sample. Randomization uses the entire data available.

While those algorithms are more robust than k-Means to initialization, we produce here a general family of algorithms that optimize Equation (2) and are even more robust to initialization.

2.1 The Expectation Maximization Type

Our first family of randomized algorithms carries out an iterative improvement as found in k-Means, Expectation Maximization, fuzzy-c-Means and k-Harmonic Means (refer to Fig. 1). The iteration alternates classifica-

```
ITERATIVE_STEP(C = {c_1, ..., c_k} ⊂ U)
    CLASSIFICATION_STEP(C)
        For j = 1, ..., k: find U_j = {u_i ∈ U | d(u_i, c_j) ≤ d(u_i, c_j')j' = 1, ..., k}
    new C ← RECONSTRUCTION_STEP
        For j = 1, ..., k: new c_j ← new estimator of median for U_j
```

Fig. 1. Body of the iteration alternating between finding a classification for the data given a model, and finding a model given classified data.

tion of data from a current model (classification step) and model refinement from classified data (reconstruction step). We say that is an Expectation Maximization type because it follows the alternation between Expectation and Maximization. That is, Expectation because in the statistical sense we estimate the hidden information (the membership to clusters of the data items). Maximization, because we use some criteria (like Maximum Likelihood) to revise the description (model) of each cluster.

Our task now is to describe the iterative algorithms that minimizes the non-crisp classification loss function with respect to a vector ω. The algorithm has the generic structure of iterative algorithms. It will start with a random set C^0. The t-th iteration proceeds as follows. First we note that non-crisp classification is computationally as costly as classification in the crisp algorithms. It essentially requires to compute SORT of the vector of distances of the data item u_i under consideration to each member of the current set of representatives C^t. This requires computations of $d(\cdot,\cdot)$ for the k representatives. Although sorting of k items requires $k \log k$ comparisons, it does not require any more dissimilarity computations and in the applications we have in mind, the computations of dissimilarity values is far more costly than the time required to sort the k values. Thus, classification is performed by a simple pass through the data and $O(nk)$ dissimilarity computations. This results in a labeling of all data items in U by a vector ranking the degrees of membership to the k clusters. That is, for each u_i, we record the rank of $d(u_i, c_j^t)$, where c_j^t is the j-th current representative. We let $\text{RANK}_i^t[j]$ denote the rank of u_i with respect to the j-th representative at the t-th classification. For example, if the first current representative is the 3rd nearest representative to u_i then $\text{RANK}_i^t[1] = 3$.

The reconstruction step computes new representatives. For each old representative, we have a degree of membership of every data item. Thus, for each $j =$

$1, \ldots, k$ we seek a new representative that minimizes $f_j^t(x) = \sum_{i=1}^n e^{\mathrm{T}}_{\mathrm{RANK}_i^t[j]} \cdot \omega d(x, u_i) = \sum_{i=1}^n \omega_{\mathrm{RANK}_i^t[j]} d(x, u_i)$. Thus, for example, if an item u_i was ranked first with respect to the j-th representative because it was its nearest representative, then the distance $d(x, u_i)$ in Equation (2) is multiplied by the largest weight in ω (which is the value in the first entry). Also, note that in the case of crisp classification, all weights are zero except the largest (and the largest weight equals 1). Thus, the minimization of $f_j^t(x)$ above just corresponds to finding the median amongst the data items in the j-th cluster. Because of this, the data item u_i such that $f_j^t(u_i)$ is smallest will be called the j-th discrete weighted median.

To detail our algorithm further, we must describe how a new representative c_j^{t+1} is computed by minimization of $f_j^t(x)$ amongst the data items u_i. The algorithm we introduce for this task is a randomized approximation inspired in a sub-quadratic randomized algorithm for computation of the discrete median [10]. For this subproblem of approximating the discrete median we note that $U = \{u_1, \ldots, u_n\}$ is the set of candidates (during the t-th iteration of the algorithm) for finding the minimum of each function $f_j^t(x)$, for $j = 1, \ldots, k$.

However, for simplicity of the notation, in what follows we drop the super-index t since it is understood we are dealing with the current iteration. We will also assume that we know which representative is being revised and we denote by OLD_MED(j) the previous approximation to the data item that minimizes $f_j(x)$ (during the t-th iteration, OLD_MED(j) is actually c_j^t).

Clearly, the discrete weighted median MED(j) can be computed in $O(\|U\|^2)$ computations of the dissimilarity $d(\cdot, \cdot)$ by simply computing $f_j(x)$ for $x = u_1, \ldots, x = u_n$ and returning the x that results in the smallest value. We will refer to this algorithm as EXHAUSTIVE. It must be used carefully because it has quadratic complexity on the size $\|U\| = n$ of the data. However, it has linear complexity of $\phi(d)$, the time to compute $d(\cdot, \cdot)$. Thus, our use of randomization.

The first step consists of obtaining a random partition of U into approximately $r = \sqrt{n}$ subsets $U_1, \ldots, U_r$ each of approximately $n/r \approx \sqrt{n}$ elements. Then, algorithm EXHAUSTIVE is applied to each of these subsets to obtain $m(j)_s = med\_d(U_s)$, $s = 1, \ldots, r$. These r items constitute candidates for the j-th discrete weighted median of U. We compute $f_j(m(j)_s)$ for $s = 1, \ldots, r$ and also $f(\text{OLD\_MED}(j))$. The item that provides the smallest amongst these (at most $r + 1$) items is returned as the new approximation to the discrete median. The algorithm has complexity $O(\phi(d)\|U\|\sqrt{\|U\|})$ because EXHAUSTIVE is applied to $\Theta(\sqrt{\|U\|})$ sets, each of size $\Theta(\sqrt{\|U\|})$; thus this requires $O(\phi(d)\|U\|\sqrt{\|U\|})$ time. Finally, $f_j(m(j)_s)$ requires $O(\phi(d)\|U\|)$ time and is performed $O(\sqrt{\|U\|})$ times. This is also $O(\phi(d)\|U\|\sqrt{\|U\|})$ time. These types of randomized algorithms for finding discrete medians have been shown mathematically and empirically to provide robust estimators of location [10]. This is because randomization is not sampling. Randomization uses the entire data available.

We enhance the fundamental results of iterative clustering algorithms by proving that our algorithm converges. We prove this by showing that both steps, the non-crisp classification step and the reconstruction step never increase the value of the objective function in Equation (2).

Lemma 1. *Let* RANK_i^t *be the rank vector for each* $u_i \in U$ $(i = 1, \ldots, n)$ *after the non-crisp classification step in the t-th iteration of the algorithm. Let*

$$M^{\omega}(C^t) = \frac{1}{\|\omega\|_1} \sum_{i=1}^{n} \sum_{j=1}^{k} \omega_{\text{RANK}_i^t[j]} d(u_i, c_j^t). \tag{3}$$

Then, the value of $\frac{1}{\|\omega\|_1} \sum_{i=1}^{n} \sum_{j=1}^{k} \omega_{\text{RANK}_i^t[j]} d(u_i, c_j^{t+1})$ *of the objective function after the reconstruction step is no larger than* $M^{\omega}(C^t)$.

Proof. First note that, by expanding the dot product $\omega^{\text{T}}\text{SORT}(\cdot)$ and using the RANK vector we have that Equation (2) and Equation (3) are the same. Then, we can reverse the order of the summation signs and also note that $1/\|\omega\|_1$ is a constant. Thus, the objective function is simply $M^{\omega}(C^t) = \frac{1}{\|\omega\|_1} \sum_{j=1}^{k} f_j^t(c_j^t)$. The reconstruction step finds new c_j^{t+1} such that $f_j^t(c_j^{t+1}) \leq f_j^t(c_j^t)$, for $j = 1, \ldots, k$ (because the previous discrete median is considered among the candidates for a new weighted discrete median). Thus,

$$M^{\omega}(C^t) \geq \frac{1}{\|\omega\|_1} \sum_{i=1}^{n} \sum_{j=1}^{k} \omega_{\text{RANK}_i^t[j]} d(u_i, c_j^{t+1}).$$

Lemma 2. *The value*

$$\begin{aligned} &M^{\omega}(C^{t+1} = \{c_1^{t+1}, \ldots, c_k^{t+1}\}) \\ &= \frac{1}{\|\omega\|_1} \sum_{i=1}^{n} \omega^{\text{T}}\text{SORT}(d(u_i, c_1^{t+1}), d(u_i, c_2^{t+1}), \ldots, d(u_i, c_k^{t+1})) \end{aligned}$$

after a classification step is no larger than $\frac{1}{\|\omega\|_1} \sum_{i=1}^{n} \sum_{j=1}^{k} \omega_{\text{RANK}_i^t[j]} d(u_i, c_j^{t+1})$ *resulting in the previous reconstruction step.*

Proof. Note that $\frac{1}{\|\omega\|_1} \sum_{j=1}^{k} f_j^t(c_j^{t+1}) = \sum_{i=1}^{n} \frac{1}{\|\omega\|_1} \sum_{j=1}^{k} \omega_{\text{RANK}_i^t[j]} d(c_j^{t+1}, u_i)$. Thus, we can say that the contribution of u_i to the objective function before the next classification is $\frac{1}{\|\omega\|_1} \sum_{j=1}^{k} \omega_{\text{RANK}_i^t[j]} d(c_j^{t+1}, u_i)$. Now, we know that after classification, this contribution by u_i is

$$\frac{1}{\|\omega\|_1} \omega^{\text{T}}\text{SORT}(d(u_i, c_1^{t+1}), d(u_i, c_2^{t+1}), \ldots, d(u_i, c_k^{t+1})).$$

Thus, the terms $d(u_i, c_j^{t+1})$ involved before and after non-crisp classification are the same, it is just that after classification they are in non-decreasing order. Since ω^{T} has entries in descending order we claim that

$$\omega^{\text{T}}\text{SORT}(d(u_i, c_1^{t+1}), d(u_i, c_2^{t+1}), \ldots, d(u_i, c_k^{t+1})) \leq \sum_{j=1}^{k} \omega_{\text{RANK}_i^t[j]} d(c_j^{t+1}, u_i)$$

whatever the order of the set $\{d(u_i, c_1^{t+1}), d(u_i, c_2^{t+1}), \ldots, d(u_i, c_k^{t+1})\}$ given by the permutation encoded in $\text{RANK}_i^t[j]$ (note the rank is the one that resulted in the previous classification, and we are about to perform the $(t+1)$-th classification).

We prove this claim by showing that if $j < j'$ and $d(u_i, c_j^{t+1}) > d(u_i, c_{j'}^{t+1})$, but $d(u_i, c_j^{t+1})$ is ranked earlier than $d(u_i, c_{j'}^{t+1})$ in the t-th ranking, then we can reduce the value of the contribution of u_i by swapping the order in the permutation of j and j'.

This is because if $j < j'$ we have $\omega_j > \omega_{j'}$ and $d(u_i, c_j^{t+1}) > d(u_i, c_{j'}^{t+1})$ implies $\omega_j(d(u_i, c_j^{t+1}) - d(u_i, c_{j'}^{t+1})) > \omega_{j'}(d(u_i, c_j^{t+1}) - d(u_i, c_{j'}^{t+1}))$.

Thus, $\omega_j d(u_i, c_j^{t+1}) + \omega_{j'} d(u_i, c_{j'}^{t+1}) > \omega_{j'} d(u_i, c_j^{t+1}) + \omega_j d(u_i, c_{j'}^{t+1}))$. Thus, swapping j and j' reduces the contribution of u_i. This proves that the sorted array of values reduces the contribution of u_i, and this is for all u_i. Thus, the new non-crisp classification produces a ranking that can not increase the value of the objective function.

Theorem 1. *Our algorithm converges.*

Proof. The domain of the objective function has size $\binom{k}{n}$, since it consists of all subsets of size k of U. Thus, the objective function has a finite range. The algorithm can not decrease the value of the objective function continuously.

This result is in contrast to the problem of the continuous median in dimensions $D \geq 2$ and the Euclidean metric (the continuous Fermat-Weber problem) where fundamental results show that it is impossible to obtain an algorithm to converge [2] (numerical algorithms usually halt because of the finite precision of digital computers). Other algorithms for non-crisp classification, like k-Harmonic Means have not been shown to converge.

2.2 The Discrete Hill-Climber Type

We now present an alternative algorithm to optimizing Equation (2). The algorithm here can be composed with the algorithm in the previous section (for example, the first can be the initialization of the latter). The result is an even more robust algorithm, still with complexity $O(n\sqrt{n})$ similarity computations.

The algorithm in this section is an interchange heuristics based on a hill-climbing search strategy. However, we require to adapt this to non-crisp classification since all previous versions [8,9,11,15,19,20,21,31] are for the crisp classification case. We will first present a quadratic-time version of our algorithm, which we will name non-crisp TaB in honor of Teitz and Bart [31] original heuristic. Later, we will use randomization to achieve subquadratic time complexity, as in the previous section.

Our algorithms explore the space of subsets $C \subset U$ of size k. Non-crisp TaB will start with a random set C^0. Then, the data items $u_i \in U$ are organized

in a circular list. Whenever the turn belonging to a data item u_i comes up, if $u_i \notin C^t$, it is used to test at most k subsets $C_j = (C^t \cup \{u_i\}) \setminus \{c_j^t\}$. That is, if u_i is currently not a representative (medoid), it is swapped with each of the k current medoids. The objective function M^ω in Equation (2) is evaluated in $M^\omega(C_j)$, for $j = 1, \ldots, k$ and if any of these values is less than the current $M^\omega(C^t)$, then the swap with the best improvement $M^\omega(C_{j_0})$ is accepted and $C^{t+1} = C_{j_0}$. In this case, or if $u_i \in C^t$ or if no C_j improves C^t, the data item u_i is placed at the end of the circular list. The turn passes to the next data item and the algorithm halts when a pass through the circular list produces no swap.

The algorithm requires $O(kn^2)$ dissimilarity evaluations because evaluating $M^\omega(C)$ requires $O(n)$ distance evaluations and at least one pass is made through the circular list to halt.

Our randomized TAB also partitions U into approximately $r = \sqrt{n}$ subsets $U_1, \ldots, U_r$ each of approximately $n/r \approx \sqrt{n}$ elements. The non-crisp TAB is applied to each of $U_1, \ldots, U_r$. Thus each U_i is clustered into k groups. This requires $O(kn\sqrt{n})$ distance calculations and results in r sets of k representatives. Then, all the resulting rk representatives are placed in a circular list. Then non-crisp TAB is applied in this list but the evaluation of $M^\omega(C)$ is performed with respect to the entire data set U. Since the circular list has length $k\sqrt{n}$, the iteration achieved by the last execution of non-crisp TAB requires $O(k^2 n\sqrt{n})$. If having k^2 in the complexity is a problem, then we can simply chose $r = \sqrt{n}/k$, and obtain an algorithm with linear complexity in k as well.

Clearly, this algorithm also converges.

3 Case Study

The literature contains many illustrations in commercial applications where clustering discovers what are the types of customers [3,4]. Recent attention for WEB Usage Mining [30] has concentrated on association rule extraction [5,23, and references]. There has been comparative less success at categorizing WEB-visitors than categorizing customers in transactional data. This WEB usage mining task is to be achieved from the visitation data to a WEB-site [29]. The goal is to identify strong correlation among users interests by grouping their navigation paths. Paths are ordered sequences of WEB pages. Many applications can then benefit from the knowledge obtained [7,16,27,28,32]. Discovery of visitor profiles is an important task for WEB-site design and evaluation [26,33]. Other examples are WEB page suggestion for users in the same cluster, pre-fetching, personalization, collaborative filtering [25] and user communities [22].

Paths are discrete structures. Several similarity measures have been defined but they all correspond to dissimilarity between high-dimensional feature vectors extracted from the paths [16,27,28,32,33]. Because the length of the path, the order of the WEB pages, the time intervals between links and many other aspects play a role in dissimilarity measures the resulting clustering problems are high dimensional. For example, a measure that has been used for WEB-path clustering is defined as follows [32]. Let $P = \{p_1, \ldots, p_m\}$ be a set of pages, and

let the corresponding usage-feature vector USAGE_{u_i} of user u_i defined by

$$\text{USAGE}_{u_i}[j] = \begin{cases} 1 & \text{if } p_j \text{ is accessed by } u_i \\ 0 & \text{Otherwise.} \end{cases}$$

Then, $\text{USAGE}(u_i, u_{i'}) = \text{USAGE}_{u_i}^T \cdot \text{USAGE}_{u_{i'}} / \|\text{USAGE}_{u_i}\| \|\text{USAGE}_{u_{i'}}\|$ is the *Usage Similarity Measure* (the cosine of the angle between the usage-feature vectors).

Clearly the dimension of the vectors involved is the number m of pages in the WEB-site, typically a number higher than 10 and usually much larger. Moreover, the Usage Similarity Measure is the simplest of the dissimilarity measures since it does not consider order, length or time along a visitation path. Other more robust dissimilarity measures are more costly to evaluate and imply feature vectors in even higher dimensions.

Also, the discrete nature of paths removes vector-based operations, like averages (means). Thus, while it is possible to compute the average of two feature vectors like $(\text{USAGE}_{u_i} + \text{USAGE}_{u_{i'}})/2$, it is not clear that the result is the feature vector of a path (a path with such feature vector may actually be infeasible given the links between pages). Also, spaces defined by dissimilarity measures are different form Euclidean spaces, since for all feature vectors $\boldsymbol{v}$, the similarity between $\boldsymbol{v}$ and itself is maximum (1); however, it is also maximum between $\boldsymbol{v}$ and any scalar transformation $\lambda \boldsymbol{v}$ of the vector itself, for all constants $\lambda > 0$.

These two challenges obstruct the use of many clustering algorithms for finding groups on WEB visitors based on their visitation paths (including k-MEANS and FUZZY-c-MEANS). The algorithms proposed to date [7,16,27,28,32] are all of quadratic time complexity (they essentially require computing the dissimilarity between all pairs of paths). These clustering efforts, although not scalable, have demonstrated the extensive benefits and sophisticated applications emerging from identifying groups of visitors to a WEB-site.

The implementation of $O(n\sqrt{n})$ time results in a dramatic improvement in CPU-time resources. Our implementation is much faster than previous matrix-based algorithms [13]. Just for the Usage dissimilarity metric the Matrix-Based algorithm requires over 18,000 CPU seconds (5 hrs!) with 590 users while our algorithms in crisp mode requires only 83 seconds (just over a minute) and in harmonic mode it requires 961 second (16 minutes). These results are on the same data set of logs identified with visitor and sessions used by Xiao et al [32]. Namely, we used the WEB-log data sets publicly available from the Boston University Computer Science Department.

To evaluate the quality of the clustering synthetic data sets are useful because it is possible to compare the results of the algorithms with the true clustering. Typically, synthetic data is generated from a mixture or from a set of k representatives by perturbing each slightly. The quality of the clustering is reflected by the proportion in which the clustering algorithm retrieves groups and identifies data items to their original group.

We reproduced to the best of our ability the synthetic generation suggested for the same task of clustering paths used by Shahabi et al [28]. Our crisp-version has already been shown to provide much better results than matrix-based al-

ternatives [13] for the usage dissimilarity measure. In what follows, we discuss results comparing our harmonic EM type algorithm ($\omega^{\mathrm{T}} = (1, 1/2, \ldots, 1/k)$ and its crisp EM type version ($\omega^{\mathrm{T}} = e_i^{\mathrm{T}}$) [14]. We used the usage-dissimilarity measure and other two measures, the frequency measure and the order measure [32]. The order measure is the same as the path measure [28]. Because our algorithms start with a random set of representatives, they were run each of them 5 times and confidence intervals are reported with 95% accuracy [14]. There are several issues we would like to comment on our results [14]. The first is that we were surprised that actually what the literature has claimed on the issue of features from paths towards dissimilarity measures is not reflected in our results. In fact, the more sophisticated the dissimilarity measure, the poorer the results. Quality was much more affected by the dissimilarity measure used than by the size of the data set or the clustering algorithm used. Our non-crisp algorithm does better, but we admit that for this data set the results are not dramatic. In fact, they are probably less impressive if one considers the CPU-time requirements [14]. The harmonic version is slower. It requires a factor of $O(k)$ more space and $O(k \log k)$ administrative work. However, the observed CPU times confirm the $O(n \log n)$ nature of our algorithms and that dissimilarity evaluation is the main cost. In fact, the usage and frequency dissimilarity functions can be computed in $O(\rho)$, where ρ is the average length of paths. However, the order dissimilarity function requires $\Omega(\rho^2)$ time to be computed.

We note that the confidence intervals for the harmonic version are smaller than for the crisp version. This indicates that harmonic is more robust to initialization. To explore further this issue we also performed on some initial experiments regarding the robustness to initialization of our algorithms. The experiment consisted of evaluating the discrepancy in clustering results between independent executions (with a different random seed) of the same algorithm. In our results [14] there is much less variance with the harmonic version. Given that the initialization is random, this suggests the algorithm finds high quality local optima with respect to its loss function. However, more thorough experimentation is required to confirm this suggestion.

We also point out that in parallel to k-Means, Expectation Maximization, and k-Harmonic Means, our harmonic EM type algorithm may place two representatives very close together attracted by the same peak in frequency. However, the theoretical foundation provided here allows also to detect this and apply the "boosting" techniques as suggested for k-Harmonic Means [34].

4 Final Remarks

We presented the theoretical framework for sub-quadratic clustering of paths with non-crisp classification. The experiments are not exhaustive [14] but they illustrate that there are benefits to be obtained with respect to quality with non-crisp classification. Moreover, they also reflect that there are trade-off to be investigated between the complexity of the dissimilarity function and its computational requirements. We indicated the possibility of hybridization between the

discrete hill-climber type and the EM type. However, because our EM-methods are generic on the vector ω they offer a range of diversity for the computational requirements in this regard. For example, one can imagine a k'-nearest neighbor classification with $k' < k$. The vector ω would have entries equal to zero from the $(k'+1)$-th entry onwards. The nonzero entries can then be a harmonic average of the k' nearest neighbors or some other combination. Thus, this is a classification that incorporates the supervised-learning technique of nearest-neighbors [6] and reduces smoothly to crisp-classification with k' closer to 1.

We have illustrated the new clustering methods with similarity measures of interests between WEB visitors. Similarity measures proposed for analysis WEB visitation [16,27,28,32,33] pose a high-dimensional non-Euclidean clustering problem. This eliminates many clustering methods. Previously [13], we (and others [18]) provided a more detailed discussion of why density-based or hierarchical methods are unsuitable to clustering paths. Our methods here are fast and robust. They are applicable to any similarity measure and can dynamically track users with high efficiency. Moreover, we have generalized fuzzy-membership or probabilistic membership to non-crisp classification.

References

1. K. Alsabti, S. Ranka, and V. Singh. An efficient k-means clustering algorithm. *IPPS 11th Int. Parallel Processing Symp.*, 1998.
2. C. Bajaj. Proving geometric algorithm non-solvability: An application of factoring polynomials. *J. of Symbolic Computation*, 2:99–102, 1986.
3. M.J.A. Berry and G. Linoff. *Data Mining Techniques — for Marketing, Sales and Customer Support.* Wiley, NY, 1997.
4. J.P. Bigus. *Data Mining with Neural Networks: Solving Business Problems from Applciation Development to Decision Support.* McGraw-Hill, NY, 1996.
5. J. Borges and M. Levene. Mining assocaition rules in hypertext databases. *4th KDD*, 149–153, NY, 1998.
6. V. Cherkassky and F. Muller. *Learning from Data.* Wiley, NY, 1998.
7. C. R. Cunha & C. F. B. Jaccound. Determining www user's next access and its application to prefetching. *Int. Symp. Computers & Communication'97*, 1997.
8. P. Densham and G. Rushton. A more efficient heuristic for solving large p-median problems. *Papers in Regional Science*, 71:307–329, 1992.
9. V. Estivill-Castro & M.E. Houle. Roboust clustering of large geo-referenced data sets. *3rd PAKDD-99*, 327–337. Springer-Verlag LNAI 1574, 1999.
10. V. Estivill-Castro and M.E. Houle. Fast randomized algorithms for robust estimation of location. *Int. Workshop on Temporal, Spatial and Spatio-Temporal Data Mining - TSDM2000, witht 4th PKDD*, 74–85, Lyon, 2000. LNAI 2007.
11. V. Estivill-Castro and A.T. Murray. Discovering associations in spatial data - an efficient medoid based approach. *2nd PAKDD-98*, 110–121, Melbourne, 1998. Springer-Verlag LNAI 1394.
12. V. Estivill-Castro and J. Yang. A fast and robust generl purpose clustering algorithm. *6th PRICAI-2000*, 208–218, Melbourne, 2000. Springer-Verlag LNAI 1886.
13. V. Estivill-Castro and J. Yang. Categorizing Visitors Dynamically by Fast and Robust Clustering of Access Logs *Asia-Pacific Conference on Web Intelligence WI-2001.* Maebashi City, Japan. 2001. N. Zhong and Y. Yao (eds) LNAI In press.

14. V. Estivill-Castro and J. Yang. Non-crisp clustering Web visitors by vast, convergent and robust algorithms on access logs. Tech. R. 2001-07, Dep. CS & SE, U. of Newcastle, `www.cs.newcastle.edu.au/Dept/techrep.html`.
15. M. Horn. Analysis and computation schemes for p-median heuristics. *Environment and Planning A*, 28:1699–1708, 1996.
16. T. Kato, H. Nakyama and Y. Yamane. Navigation analysis tool based on the correlation between contents and access patterns. Manuscript. `http://citeseer.nj.nec.com/354234.html`.
17. M Lorr. *Cluster Analysis for Social Scientists.* Jossey-Bass, San Francisco, 1983.
18. T. Morzy, M. Wojciechowski, and Zakrzewicz. Scalabale hierarchical clustering methods for sequences of categorical values. D. Cheung, et al eds., *5th PAKDD*, 282–293, Hong Kong, 2001. LNAI 2035.
19. A.T. Murray. Spatial characteristics and comparisons of interaction and median clustering models. *Geographical Analysis*, 32:1–, 2000.
20. A.T. Murray and R.L. Church. Applying simulated annealing to location-planning models. *J. of Heuristics*, 2:31–53, 1996.
21. R.T. Ng and J. Han. Efficient and effective clustering methods for spatial data mining. *20th VLDB*, 144–155, 1994. Morgan Kaufmann.
22. G. Paliouras, C. Papatheodorou, V. Karkaletsis, and C. Spyropoulos. Clustering the users of large web sites into communities. P. Langley, ed., *17th Int. Conf. on Machine Learning*, 719–726, 2000. Morgan Kaufmann.
23. J. Pei, J. Han, B. Mortazavi-asl, and H. Zhu. Mining access patterns efficiently from web logas. *4th PAKDD*, 396–407, 2000. Springer-Verlag LNCS 1805.
24. D. Pelleg and A. Moore. X-means: Extending K-means with efficient estimation of the number of clusters. In P. Langley, ed., *17th Int. Conf. Machine Learning*, 727–734, CA, 2000. Morgan Kaufmann.
25. D.M. Pennock, E. Horvitz, S. Lawrence, and C.L. Giles. Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach. *16th Conf. on Uncertanity in Artificial Intelligence*, 473–840, 2000. Morgan Kaufmann.
26. M Perkowitz and O. Etzioni. Adaptive web sites: an AI challenge. *IJCAI*, 16–23, Nagoya, 1998.
27. M Perkowitz and O. Etzioni. Adaptive web sites: Automatically synthesizing web pages. *15th National C. on AI*, 727–732, Madison, July 1998. AAAI Press.
28. C. Shahabi, A. M. Zarkesh, J. Adibi, and V. Shah. Knowledge discovery from users web page navigation. *IEEE RIDE'97*, 20–31, 1997.
29. M. Spiliopoulou. Web usage mining for web site evaluation. *C. of the ACM*, 43:127–134, 2000.
30. J. Srivastava, R. Cooley, M. Deshpande, and P.-N. Tan. Web usage mining: Discovery and applications of usage patterns from web data. *SIGKDD Esplorations*, 1(2):12–23, January 2000.
31. M.B. Teitz and P. Bart. Heuristic methods for estimating the generalized vertex median of a weighted graph. *Operations Research*, 16:955–961, 1968.
32. J. Xiao, Y. Zhang, X. Jia, and T. Li. Measuring similarity of interests for clustering web-users. *12th ADC 2001*, 107–114, Gold Coast, IEEE Computer Society.
33. A.M. Zarkesh, J. Adibi, C. Shahabi, R. Sadri, and V. Shah. Analysis and design of server informative WWW-sites. *6th CIKM*, 254–261, Las Vegas, 1997. ACMPress.
34. B. Zhang, M. Hsu, and U. Dayal. K-harmonic means — a spatial clustering algorithm with boosting. *Int. Workshop on Temporal, Spatial and Spatio-Temporal Data Mining - TSDM2000, with 4th PKDD*, 31–42, Lyon, 2000. LNAI 2007.