Self-Organizing Maps of Relatonal Data Based On Multiple Dissimilarity Matrices

Francisco de A.T. de Carvalho [a,*], Anderson B. S. Dantas [a] and Yves Lechevallier [b]

[a] *Centro de Informática, Universidade Federal de Pernambuco, Av. Prof. Luiz Freire, s/n - Cidade Universitária - CEP 50740-540 - Recife (PE) - Brazil*

[b] *INRIA-Institut National de Recherche en Informatique et en Automatique Domaine de Voluceau-Rocquencourt B. P.105, 78153 Le Chesnay Cedex, France*

Abstract

This paper introduces hard clustering algorithms that are able to partition objects taking into account simultaneously their relational descriptions given by multiple dissimilarity matrices. These matrices have been generated using different sets of variables and dissimilarity functions. These methods are designed to furnish a partition and a prototype for each cluster as well as to learn a relevance weight for each dissimilarity matrix by optimizing an adequacy criterion that measures the fitting between the clusters and their representatives. These relevance weights change at each algorithm iteration and can either be the same for all clusters or different from one cluster to another. Experiments with data sets (synthetic and from UCI machine learning repository) described by real-valued variables as well as with time trajectory data sets show the usefulness of the proposed algorithms.

Key words: Self-organizing Maps, Partitioning Clustering Algorithms, Relational Data, Relevance Weight, Multiple Dissimilarity Matrices.

* Corresponding Author. tel.: +55-81-21268430 ; fax:+55-81-21268438
Email addresses: fatc@cin.ufpe.br (Francisco de A.T. de Carvalho), absd@cin.ufpe.br (Anderson B. S. Dantas), Yves.Lechevallier@inria.fr (Yves Lechevallier).

[1] Acknowledgements. This research was partially supported by grants from CNPq, FACEPE (Brazilian Agencies) and from a conjoint research project FACEPE and INRIA (France).

1 Introduction

Clustering methods organize a set of items into clusters such that items within a given cluster have a high degree of similarity, whereas those of different clusters have a high degree of dissimilarity. These methods have been widely applied in fields such as taxonomy, image processing, information retrieval and data mining. The most popular clustering techniques are hierarchical and partitioning methods [1,2].

Hierarchical methods yield complete hierarchy, i.e., a nested sequence of partitions of the input data. Hierarchical methods can be agglomerative [3–7] or divisive [8–12]. Agglomerative methods yield a sequence of nested partitions starting with trivial clustering in which each item is in a unique cluster and ending with clustering in which all items are in the same cluster. A divisive method starts with all items in a single cluster and performs a splitting procedure until a stopping criterion is met (usually upon obtaining a partition of singleton clusters).

Partitioning methods seek to obtain a single partition of the input data into a fixed number of clusters. These methods often look for a partition that optimizes (usually locally) an objective function. To improve cluster quality, the algorithm is run multiple times with different starting points and the best configuration obtained from the total runs is used as the output clustering. Partitioning methods can be divided into hard clustering [13–17] and fuzzy clustering [18–22]. Hard clustering furnishes a partition in which each object of the data set is assigned to one and only one cluster. Fuzzy clustering generates a fuzzy partition that furnishes a degree of membership of each pattern in a given cluster. This gives the flexibility to express that objects belong to more than one cluster at the same time.

There are two common representations of the objects upon which clustering can be based : feature data and relational data. When each object is described by a vector of quantitative or qualitative values, the set of vectors describing the objects is called a *feature data*. Alternatively, when each pair of objects is represented by a relationship, then it is called *relational data*. The most common case of relational data is when one has (a matrix of) dissimilarity data, say $R = [r_{kl}]$, where r_{kl} is the pairwise dissimilarity (often a distance) between objects k and l. Clustering of relational data is very useful when the objects cannot be described by a vector of feature values, when the distance measure does not have a closed form, etc [10], [23–26]. Recently, Frigui et al [27] proposed CARD, a relational fuzzy clustering algorithm that is able to partition objects taking into account multiple dissimilarity matrices and that learns a relevance weight for each dissimilarity matrix in each cluster. CARD is mainly based on the well-known fuzzy clustering algorithms for relational data NERF

[26] and FANNY [10]. As remarked by [27], several applications can benefit from relational clustering algorithms based on multiple dissimilarity matrices. In image data base categorization, the relationship among the objects may be described by multiple dissimilarity matrices and the most effective dissimilarity measures do not have a closed form or are not differentiable with respect to prototype parameters.

This paper extends the dynamic hard clustering algorithm for relational data [23], [24], into hard clustering algorithms that are able to partition objects taking into account simultaneously their relational descriptions given by multiple dissimilarity matrices. The main idea is to obtain a collaborative role of the different dissimilarity matrices [28] to obtain a final partition. These dissimilarity matrices could have been generated using different sets of variables and a fixed dissimilarity function (in this case, the final partition is given according to different views (i.e., different sets of variables) describing the objects), or using a fixed set of variables and different dissimilarity functions (in this case, the final partition is given according to different dissimilarity functions) or using different sets of variables and dissimilarity functions. As pointed out by [27], the influence of the different dissimilarity matrices can not be equally important in the definition of the clusters in the final partition. Thus, to obtain a meaningful partition from all dissimilarity matrices, the relational hard clustering algorithms given in this paper are designed to give a partition and a prototype for each cluster as well as to learn a relevance weight for each dissimilarity matrix by optimizing an adequacy criterion that measures the fitting between the clusters and their representatives. These relevance weights change at each algorithm's iteration and can either be the same for all clusters or different from one cluster to another.

This paper is organized as follows. Section **??** first reviews a partitioning dynamic hard clustering algorithm based on a single dissimilarity matrix (section **??**) and then introduces partitioning dynamic hard clustering algorithms based on multiple dissimilarity matrices with relevance weight for each dissimilarity matrix either estimated locally (section **??**) or estimated globally (section **??**). Section 4 gives empirical results to show the usefulness of these relational clustering algorithms. Finally, section 5 gives final remarks and comments.

2 Introduction to the Self-Organizing Maps

2.1 Self-Organizing Maps for Object Data

2.1.1 Self-Organizing Map (SOM)

Kohonen's SOM [**?**] is used nowadays through numerous domains and has been successfully applied in numerous applications. It is a very popular tool used for visualizing high dimensional data spaces. SOM can be considered as doing vector quantization and/or clustering while preserving the spatial ordering of the input data reflected by implementing an ordering of the prototype vectors (also called codebook vectors, cluster centroids or referent vectors) in a one or two dimensional output space. The SOM consists of neurons organized on a regular low-dimensional grid, called the map. More formally, the map is described by a graph (C, Γ). C is a set of m interconnected neurons having a discrete topology defined by Γ. For each pair of neurons (c, r) on the map, $\delta(c, r)$ is defined as the distance function between c and r on the graph. This distance imposes a neighborhood relation between neurons. Each neuron c is represented by a p-dimensional prototype $\mathbf{w}_c = (w_c^1, ..., w_c^p)$, where p is equal to the dimension of the input vectors.

Let $E = \{1, \ldots, n\}$ the set of objects, where each object $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})$ $(i = 1, \ldots, n)$ belongs to $\mathbb{R}^p$.

2.2 The Algorithm

SOM basic algorithm performs the following steps:

1) *Initialization.*
   Fix the number m of neurons (clusters);
   Fix $h_{\delta(j,l)}(0)$ (start neighborhood function, where $\delta(j, l)$ is a fixed distance function between neurons j and l);
   Fix the kernel fuction K;
   Fix the number of iterations N_{iter};
   Fix $\eta(0)$ (start learning rate);
   Set $t \leftarrow 1$;
   Randomly select m distinct prototypes $\mathbf{w}_c^{(0)} \in E$ $(c = 1, \ldots, m)$;
   Set the map $L(m, \mathbf{W}^0)$, where $\mathbf{W}^0 = (\mathbf{w}_1^{(0)}, \ldots, \mathbf{w}_m^{(0)})$;
2) *Step 1: Sampling.*
   Grab a random input vector $\mathbf{x}_i(t)$;
3) *Step 2: Selection.*
   Find the best Kohonen neuron (winner) $\mathbf{w}_c(t)$ in relation to minimum Eu-

clidian distance rule:

$$f(\mathbf{x}_i(t)) = \min_{1 \leq j \leq m} \sum_{k=1}^{p} (x_{ik}(t) - w_{jk}(t))^2;$$

4) *Step 3: Weight modification.*
For all neurons j within a specified neighborhood radius of the winning neuron $\mathbf{w}_c$, adjust the weights according to the following formula:

$$\mathbf{w}_j(t+1) = \mathbf{w}_j(t) + \eta(t) h_{\delta(\mathbf{w}_c, w_j)}(t)(\mathbf{x}_i(t) - \mathbf{w}_j(t));$$

5) *Updating.*
Update both learning rate $\eta(t)$ and neighborhood function $h_{l,j}(t)$
6) *Stopping criterion.*
If $t = N_{Iter}$, stop; otherwise, go to 2 (Step 1).

2.2.1 A Batch Self-Organizing Map (BSOM)

This section presents the batch self-organizing map (BSOM) introduced by [**?**]. Let $E = \{1, \ldots, n\}$ the set of objects, where each object $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})$ $(i = 1, \ldots, n)$ belongs to $\mathbb{R}^p$. Each neuron on the map is represented by a prototype $\mathbf{w}_c = (w_{c1}, \ldots, w_{cp})$ $(c = 1, \ldots, m)$ that also belongs to $\mathbb{R}^p$.

The batch SOM training algorithm [**?**] is an iterative two-step algorithm (affectation and representation steps, discussed later) in which the whole data set (noted E) is presented to the map before any adjustments are made. The batch SOM algorithm [**?**] minimizes the following cost function:

$$J = \sum_{i=1}^{n} \sum_{r=1}^{m} K^T(\delta(f^T(\mathbf{x}_i), r)) d^2(\mathbf{x}_i, \mathbf{w}_r) \tag{1}$$

where f is the allocation function and $f(\mathbf{x}_i)$ stands for the neuron of the map that is associated to the object $\mathbf{x}_i$, and $\delta(f(\mathbf{x}_i), r))$ is the distance on the map between a neuron r and the neuron that is allocated to the object $\mathbf{x}_i$. Moreover, K^T, that is parameterized by T (where T stands for temperature), is the neighborhood kernel function that defines influence region around each neuron r.

This cost function is an extension of the *k-means* cost function, where the Euclidean distance is replaced by a generalized distance:

$$d^T(\mathbf{x}_i, \mathbf{w}_{f(\mathbf{x}_i)}) = \sum_{r=1}^{m} K^T(\delta(f^T(\mathbf{x}_i), r)) d^2(\mathbf{x}_i, \mathbf{w}_r) \tag{2}$$

where

$$d^2(\mathbf{x}_i, \mathbf{w}_r) = \sum_{j=1}^{p} (x_{ij} - w_{rj})^2 \tag{3}$$

is the Euclidean distance. This generalized distance is a weighted sum of the euclidean distances between $\mathbf{x}_i$ and all the reference vectors of the neighborhood of the neuron $f(\mathbf{x}_i)$. It takes into account all the neurons of the map.

When T is kept fixed, the minimization of J is performed iteratively in two steps: affectation and representation.

During the affectation step, the reference vectors (prototypes) are kept fixed. The cost function J is minimized with respect to the allocation function and each individual $\mathbf{x}_i$ is assigned to its nearest neuron:

$$c = f^T(\mathbf{x}_i) = arg \min_{1 \leq r \leq m} d^T(\mathbf{x}_i, \mathbf{w}_r) \tag{4}$$

During the representation step, the allocation function is kept fixed. The cost function J is minimized with respect to the prototypes. The prototype $\mathbf{w}_c$ is updated for each neuron according to:

$$\mathbf{w}_c = \frac{\sum_{i=1}^{n} K^T(\delta(f^T(\mathbf{x}_i), c))\mathbf{x}_i}{\sum_{i=1}^{n} K^T(\delta(f^T(\mathbf{x}_i), c))}. \tag{5}$$

2.3 The Algorithm

The batch SOM algorithm for a usual cooling schedule can be summarized as follows:

1) *Initialization.*
   Fix the number m of neurons (clusters);
   Fix δ; Fix the kernel fuction K^T;
   Fix the number of iterations N_{iter};
   Fix T_{min}, T_{max}; Set $T \leftarrow T_{max}$; Set $t \leftarrow 0$;
   Randomly select m distinct prototypes $\mathbf{w}_c^{(0)} \in E\,(c = 1, \ldots, m)$;
   Set the map $L(m, \mathbf{W}^0)$, where $\mathbf{W}^0 = (\mathbf{w}_1^{(0)}, \ldots, \mathbf{w}_m^{(0)})$;
   Assign each object $\mathbf{x}_i$ to the closest neuron (cluster) according to equation (4);
2) *Step 1: Representation.*
   Set $T = T_{max}(\frac{T_{min}}{T_{max}})^{\frac{t}{N_{iter}-1}}$;

The allocation function is kept fixed;
Compute the prototypes $\mathbf{w}_c^{(t)}$ $(c = 1, \ldots, m)$ according to equation (5);

3) *Step 2: Affectation.*
The prototypes $\mathbf{w}_c^{(t)}$ $(c = 1, \ldots, m)$ are fixed. Assign each individual $\mathbf{x}_i$ $(i = 1, \ldots, n$ to its nearest neuron according to equation (4);

4) *Stopping criterion.*
If $T = T_{min}$ then STOP; otherwise set $t = t + 1$ and go to 2 (Step 1).

2.4 Self-Organizing Maps for Relational Data

2.4.1 Batch Self-Organizing Map for dissimilarity data

The batch SOM algorithm for dissimilarity data (B-SOM for dissimilarity data) introduced by [**?**] stems from the original SOM as described above. The main difference is on the data to be classified. These data are represented by a dissimilarity relationship. Let $E = \{e_1, \ldots, e_n\}$ be a set of n objects and a dissimilarity measure $d(e_i, e_l)$ between the objects e_i and e_l. Each neuron c is represented by a reference vector (or prototype) $g_c = e_j$, $e_j \in E$. In classical SOM each referent vector evolves in the entire input space $\mathbb{R}^p$, in this approach each neuron has a finite number of representations.

The batch training algorithm is an iterative two-step algorithm (affectation and representation steps, discussed later) in which the whole set of input (noted E) is presented to the map before any adjustments be achieved. During learning the following cost function is minimized:

$$J = \sum_{i=1}^{n} \sum_{r=1}^{m} K^T(\delta(f(e_i), r)) d(e_i, g_r) \tag{6}$$

where f is the allocation function and $f(e_i)$ stands for the neuron of the map that is associated to the object e_i and $\delta(f(e_i), r)$ is the distance on the map between a neuron r and the neuron that is allocated to the object e_i. The concept of neighborhood is taken into account through kernel functions K which are positive and such that $\lim_{|x|\to\infty} K(x) = 0$ [**?**]. Moreover, K^T that is parameterized by T (where T stands for temperature), is the neighborhood kernel function that defines the influence region around each neuron r. The smaller the value of T, the fewer the neurons that belong to the neighborhood of a given neuron r.

This cost function is an extension of the *k-means* cost function, where the Euclidean distance is replaced by a generalized distance denoted d^T:

$$d^T(e_i, g_{f(e_i)}) = \sum_{r=1}^{m} K^T(\delta(f^T(e_i), r))d(e_i, g_r) \tag{7}$$

This generalized distance is a weighted sum of the euclidean distances between e_i and all the prototypes of the neighborhood of the neuron $f(e_i)$.

When T is kept fixed, the minimization of J is performed iteratively in two steps: affectation and representation.

During affectation step, the function f associates each element e_i to the neuron whose referent vector is "closest" to e_i and decreases the clustering criterion J. Each individual e_i is assigned to its nearest neuron:

$$c = f^T(e_i) = arg \min_{1 \leq r \leq m} d^T(e_i, g_r) \tag{8}$$

During the representation step, new prototypes representing each cluster are selected. The prototype g_r^* of the cluster C_r, which minimizes the cost function J is determined by the equation:

$$g_r^* = arg \min_{e \in E} \sum_{i=1}^{n} K^T(\delta(f^T(e_i), r))d^T(e_i, e_r) \tag{9}$$

2.5 The B-SOM Algorithm

(1) Initialization.
Fix the number m of neurons (clusters);
Fix δ; Fix the kernel function K^T;
Fix the number of iterations N_{iter};
Fix T_{min}, T_{max}; Set $T \leftarrow T_{max}$; Set $t \leftarrow 0$;
Randomly select m distinct prototypes $g_c^{(0)} \in E(c = 1, \ldots, m)$;
Set the map $L(m, G^0)$, where $G^0 = (g_1^{(0)}, \ldots, g_m^{(0)})$;
Assign each object e_i to the closest neuron (cluster) according to equation 8;

(2) Step 1: Representation.
Set $T = T_{max} * (\frac{T_{min}}{T_{max}})^{\frac{t}{N_{iter}-1}}$;
The allocation function is kept fixed;
Select the prototypes $g_c^{(t)}(c = 1, \ldots, m)$ according to equation 9;

(3) Step 2: Affectation.
The prototypes $g_c^{(t)}(c = 1, \ldots, m)$ are fixed. Assign each individual $e_i(i =$

$1, \ldots, n$) to its nearest neuron according to equation 8;

(4) Stopping criterion. If $T = T_{min}$ then Stop; otherwise set $t = t + 1$ and go to 2 (Step 1).

3 Self-Organizing Maps for Relational Data Based On Multiple Dissimilarity Matrices

Let $E = \{e_1, \ldots, e_n\}$ be the set of n objects and let p dissimilarity matrices $\mathbf{D}_j = [d_j(e_i, e_l)]\,(j = 1, \ldots, p)$, where $d_j(e_i, e_l)$ gives the dissimilarity between objects e_i and $e_l\,(i, l = 1, \ldots, n)$ on dissimilarity matrix $\mathbf{D}_j$. A particularity of this method is that it assumes that the prototype G_k of cluster C_k is a subset of fixed cardinality $1 \leq q << n$ of the set of objects E (even if, for a matter of simplicity, very often $q = 1$), i.e., $G_k \in E^{(q)} = \{A \subset E : |A| = q\}$.

The batch SOM algorithm for relational data based on multiple dissimilarity matrices introduced in this paper is an iterative three-step algorithm (representation, weighting and affectation steps) in which the whole data set is presented to the map before that any adjustments are made.

The cost function of this batch SOM algorithm is given by:

$$J = \sum_{e_i \in E} \sum_{l=1}^{c} K^T(\delta(\chi(e_i), l)) D_{\boldsymbol{\lambda}_l}(e_i, G_l) \tag{10}$$

in which $D_{\boldsymbol{\lambda}_l}$ is the global matching between an example $e_i \in P_l$ and the cluster prototype $G_l \in E^{(q)}$, parameterized by the relevance weight vector $\boldsymbol{\lambda}_l = (\lambda_{l1}, \ldots, \lambda_{lp})$ of the dissimilarity matrices $\mathbf{D}_j$ into cluster $P_l\,(l = 1, \ldots, c)$.

According to the matching function, there are different batch SOM algorithms. Here, we consider matching functions with relevance weight for each dissimilarity matrix estimated either locally or globally.

The main idea of the matching functions which are defined by a vector of weights in which the weights are estimated locally for each cluster is to compare clusters and their prototypes using a different matching measure associated with each cluster that changes at each iteration, i.e., the distance is not determined absolutely and is different from one cluster to another.

Two matching functions with relevance weight for each dissimilarity matrix estimated locally are considered depending on whether the sum of weights is equal to one or the product of the weights is equal to one. They are:

a) Matching function parameterized by the vector of relevance weights $\boldsymbol{\lambda}_l = (\lambda_{l1}, \ldots, \lambda_{lp})$, in which $\lambda_{lj} > 0$ and $\prod_{j=1}^{p} \lambda_{lj} = 1$, and associated with cluster $P_l \, (l = 1, \ldots, c)$

$$D_{\boldsymbol{\lambda}_l}(e_i, G_l) = \sum_{j=1}^{p} \lambda_{lj} D_j(e_i, G_l) = \sum_{j=1}^{p} \lambda_{lj} \sum_{e \in G_l} d_j(e_i, e) \tag{11}$$

b) Matching function parameterized by both the parameter s and the vector of relevance weights $\boldsymbol{\lambda}_l = (\lambda_{l1}, \ldots, \lambda_{lp})$, in which $1 < s < \infty$, $\lambda_{lj} > 0$ and $\sum_{j=1}^{p} \lambda_{lj} = 1$, and associated with cluster $P_l \, (l = 1, \ldots, c)$

$$D_{(\boldsymbol{\lambda}_l, s)}(e_i, G_l) = \sum_{j=1}^{p} (\lambda_{lj})^s D_j(e_i, G_l) = \sum_{j=1}^{p} (\lambda_{lj})^s \sum_{e \in G_l} d_j(e_i, e) \tag{12}$$

In equations (11) and (12), $D_j(e_i, G_l) = \sum_{e \in G_l} d_j(e_i, e)$ is the local dissimilarity between an example $e_i \in P_l$ and the cluster prototype $G_l \in E^{(q)}$ on dissimilarity matrix $\mathbf{D}_j \, (j = 1, \ldots, p)$.

The main idea of the matching functions which are defined by a vector of weights in which the weights are estimated globally for all clusters at once is that there is a distance to compare clusters and their prototypes that changes at each iteration but is the same for all clusters.

Two matching functions with relevance weight for each dissimilarity matrix estimated globally are considered depending on whether the sum of weights is equal to one or the product of the weights is equal to one. They are:

c) Matching function parameterized by the vector of relevance weights $\boldsymbol{\lambda}_l = \boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_p) \, (l = 1, \ldots, c)$, in which $\lambda_j > 0$ and $\prod_{j=1}^{p} \lambda_j = 1$

$$D_{\boldsymbol{\lambda}}(e_i, G_l) = \sum_{j=1}^{p} \lambda_j D_j(e_i, G_l) = \sum_{j=1}^{p} \lambda_j \sum_{e \in G_l} d_j(e_i, e) \tag{13}$$

d) Matching function parameterized by both the parameter s and the vector of relevance weights $\boldsymbol{\lambda}_r = \boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_p) \, (l = 1, \ldots, c)$, in which $1 < s < \infty$, $\lambda_j > 0$ and $\sum_{j=1}^{p} \lambda_j = 1$

$$D_{(\boldsymbol{\lambda}, s)}(e_i, G_l) = \sum_{j=1}^{p} (\lambda_j)^s D_j(e_i, G_l) = \sum_{j=1}^{p} (\lambda_j)^s \sum_{e \in G_l} d_j(e_i, e) \tag{14}$$

In equations (13) and (14), $D_j(e_i, G_l)$ is again the local dissimilarity between an example $e_i \in P_l$ and the cluster prototype $G_l \in E^{(q)}$ on dissimilarity matrix $\mathbf{D}_j \, (j = 1, \ldots, p)$.

When T is kept fixed, the minimization of J is performed iteratively in three steps: representation, weighting and affectation.

3.1 Representation step: computation of the best prototypes

During the representation step, the partition $P^{(t-1)} = (P_1^{(t-1)}, \ldots, P_c^{(t-1)})$ and the vectors of weights ${\boldsymbol{\lambda}_l}^{(t-1)}\,(r = 1, \ldots, c)$ are kept fixed. The cost function J is minimized with respect to the prototypes.

Proposition 3.1 *The prototype is updated for each neuron according to the matching function used:*

(1) If the matching function is given by equation (11), compute the prototype $G_l^{(t)} = G^* \in E^{(q)}$ of cluster $P_l^{(t-1)}$ $(l = 1, \ldots, c)$ according to:

$$G^* = argmin_{G \in E^{(q)}} \sum_{e_i \in E} K^T(\delta(\chi^{(t-1)}(e_i), l)) \sum_{j=1}^{p} \lambda_{lj}^{(t-1)} \sum_{e \in G} d_j(e_i, e) \quad (15)$$

(2) If the matching function is given by equation (12), compute the prototype $G_l^{(t)} = G^* \in E^{(q)}$ of cluster $P_l^{(t-1)}$ $(r = 1, \ldots, c)$ according to:

$$G^* = argmin_{G \in E^{(q)}} \sum_{e_i \in E} K^T(\delta(\chi^{(t-1)}(e_i), l)) \sum_{j=1}^{p} (\lambda_{lj}^{(t-1)})^s \sum_{e \in G} d_j(e_i, e) \quad (16)$$

(3) If the matching function is given by equation (13), compute the prototype $G_l^{(t)} = G^* \in E^{(q)}$ of cluster $P_l^{(t-1)}$ $(r = 1, \ldots, c)$ according to:

$$G^* = argmin_{G \in E^{(q)}} \sum_{e_i \in E} K^T(\delta(\chi^{(t-1)}(e_i), l)) \sum_{j=1}^{p} \lambda_j^{(t-1)} \sum_{e \in G} d_j(e_i, e) \quad (17)$$

(4) Finally, if the matching function is given by equation (14), compute the prototype $G_l^{(t)} = G^* \in E^{(q)}$ of cluster $P_l^{(t-1)}$ $(l = 1, \ldots, c)$ according to:

$$G^* = argmin_{G \in E^{(q)}} \sum_{e_i \in E} K^T(\delta(\chi^{(t-1)}(e_i), l)) \sum_{j=1}^{p} (\lambda_j^{(t-1)})^s \sum_{e \in G} d_j(e_i, e) \quad (18)$$

3.2 Weighting step: definition of the best vectors of weights

During the weighting step, the partition $P^{(t-1)} = (P_1^{(t-1)}, \ldots, P_c^{(t-1)})$ and the prototypes $G_l^{(t)} \in E^{(q)}\,(l = 1, \ldots, c)$ are fixed. The cost function J is minimized with respect to the vectors of weights.

Proposition 3.2 *The vectors of weights are updated according to the matching function used:*

(1) If the matching function is given by equation (11), the vectors of weights $\boldsymbol{\lambda}_l^{(t)} = (\lambda_{l1}^{(t)}, \ldots, \lambda_{lp}^{(t)})\,(l = 1, \ldots, c)$*, under* $\lambda_{lj}^{(t)} > 0$ *and* $\prod_{j=1}^{p} \lambda_{lj}^{(t)} = 1$*, have their weights* $\lambda_{lj}^{(t)}\,(j = 1, \ldots, p)$ *calculated according to:*

$$\lambda_{lj}^{(t)} = \frac{\left\{ \prod_{h=1}^{p} \left[\sum_{e_i \in E} K^T(\delta(\chi^{(t-1)}(e_i), l)) \sum_{e \in G_l^{(t)}} d_h(e_i, e) \right] \right\}^{\frac{1}{p}}}{\left[\sum_{e_i \in E} K^T(\delta(\chi^{(t-1)}(e_i), l)) \sum_{e \in G_l^{(t)}} d_j(e_i, e) \right]} \tag{19}$$

(2) If the matching function is given by equation (12), the vectors of weights $\boldsymbol{\lambda}_l^{(t)} = (\lambda_{l1}^{(t)}, \ldots, \lambda_{lp}^{(t)})\,(l = 1, \ldots, c)$*, under* $\lambda_{lj}^{(t)} > 0$ *and* $\sum_{j=1}^{p} \lambda_{lj}^{(t)} = 1$*, have their weights* $\lambda_{lj}^{(t)}\,(j = 1, \ldots, p)$ *calculated according to:*

$$\lambda_{lj}^{(t)} = \left[\sum_{h=1}^{p} \left(\frac{\sum_{e_i \in E} K^T(\delta(\chi^{(t-1)}(e_i), l)) \sum_{e \in G_l^{(t)}} d_j(e_i, e)}{\sum_{e_i \in E} K^T(\delta(\chi^{(t-1)}(e_i), l)) \sum_{e \in G_l^{(t)}} d_h(e_i, e)} \right)^{\frac{1}{s-1}} \right]^{-1} \tag{20}$$

(3) If the matching function is given by equation (13), the vector of weights $\boldsymbol{\lambda}^{(t)} = (\lambda_1^{(t)}, \ldots, \lambda_p^{(t)})$ *, under* $\lambda_j^{(t)} > 0$ *and* $\prod_{j=1}^{p} \lambda_j^{(t)} = 1$*, have their weights* $\lambda_j^{(t)}\,(j = 1, \ldots, p)$ *calculated according to:*

$$\lambda_j^{(t)} = \frac{\left\{ \prod_{h=1}^{p} \left[\sum_{r=1}^{c} \left(\sum_{e_i \in E} K^T(\delta(\chi^{(t-1)}(e_i), l)) \sum_{e \in G_l^{(t)}} d_h(e_i, e)] \right) \right] \right\}^{\frac{1}{p}}}{\sum_{r=1}^{c} \left[\sum_{e_i \in E} K^T(\delta(\chi^{(t-1)}(e_i), l)) \sum_{e \in G_l^{(t)}} d_j(e_i, e) \right]} \tag{21}$$

(4) If the matching function is given by equation (14), compute the vector of weights $\boldsymbol{\lambda}^{(t)} = (\lambda_1^{(t)}, \ldots, \lambda_p^{(t)})$ *, under* $\lambda_j^{(t)} > 0$ *and* $\sum_{j=1}^{p} \lambda_j^{(t)} = 1$*, have their weights* $\lambda_j^{(t)}\,(j = 1, \ldots, p)$ *calculated according to:*

$$\lambda_j^{(t)} = \left[\sum_{h=1}^{p} \left(\frac{\sum_{r=1}^{c} \sum_{e_i \in E} K^T(\delta(\chi^{(t-1)}(e_i), l)) \sum_{e \in G_l^{(t)}} d_j(e_i, e)}{\sum_{r=1}^{c} \sum_{e_i \in E} K^T(\delta(\chi^{(t-1)}(e_i), l)) \sum_{e \in G_l^{(t)}} d_h(e_i, e)} \right)^{\frac{1}{s-1}} \right]^{-1} \tag{22}$$

Proof

Here we give the proof for the case where the matching function is given by equation (11). The proof for the remaining cases are obtained in a similar way.

As the partition $P = (P_1, \ldots, P_c)$ of E into c clusters and the prototypes $G_1, \ldots, G_c$ are fixed, one can rewrite the cost function J as:

$$J(\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_c) = \sum_{l=1}^{c} J_l(\boldsymbol{\lambda}_l)$$

with

$$J_l(\boldsymbol{\lambda}_l) = J_l(\lambda_{l1}, \ldots, \lambda_{lp}) = \sum_{j=1}^{p} \lambda_{lj}\, J_{lj} \text{ where } J_{lj} = \sum_{e_i \in E} K^T(\delta(\chi(e_i), l)) \sum_{e \in G_l} d_j(e_i, e)$$

Let $g(\lambda_{l1}, \ldots, \lambda_{lp}) = \lambda_{l1} \times \ldots \times \lambda_{lp} - 1$. One can determine the extremes of $J_l(\lambda_{l1}, \ldots, \lambda_{lp})$ with the restriction $g(\lambda_{l1}, \ldots, \lambda_{lp}) = 0$. From the Lagrange multiplier method, and after some algebra, it follows that (for $j = 1, \ldots, p$)

$$\lambda_{lj} = \frac{(\prod_{h=1}^{p} J_{lh})^{1/p}}{J_{lj}} = \frac{\left\{\prod_{h=1}^{p} \left(\sum_{e_i \in E} K^T(\delta(\chi(e_i), l)) \sum_{e \in G_l} d_h(e_i, e)\right)\right\}^{\frac{1}{p}}}{\sum_{e_i \in E} K^T(\delta(\chi(e_i), l)) \sum_{e \in G_l} d_j(e_i, e)}$$

Thus, an extreme value of J_l is reached when $J_l(\lambda_{l1}, \ldots, \lambda_{lp}) = p\{J_{l1} \times \ldots \times J_{lp}\}^{1/p}$. As $J_l(1, \ldots, 1) = \sum_{j=1}^{p} J_{lj} = J_{l1} + \ldots + J_{lp}$ and as it is well known that the arithmetic mean is greater than the geometric mean, i.e., $\frac{1}{p}(J_{l1} + \ldots + J_{lp}) > \{J_{l1} \times \ldots \times J_{lp}\}^{1/p}$ (the equality holds only if $J_{l1} = \ldots = J_{lp}$), one can conclude that this extreme is a minimum value.

3.3 Affectation step: definition of the best partition

During the affectation step, the reference vectors (prototypes) $G_l^{(t)} \in E^{(q)}$ $(l = 1, \ldots, c)$ and the vectors of weights $\boldsymbol{\lambda}_l{}^{(t-1)}$ $(r = 1, \ldots, c)$ are kept fixed. The cost function J is minimized with respect to the allocation function.

Proposition 3.3 *Each individual $e_i \in E$ is assigned to its nearest neuron according to the matching function used :*

(1) If the matching function is given by equation (11), assign individual $e_i \in$

E to cluster C_m according to:

$$m = (\chi^{(t)}(e_i))^{(t)} = argmin_{1 \leq r \leq c} \sum_{l=1}^{c} K^T(\delta(r,l)) \sum_{j=1}^{p} \lambda_{lj}^{(t)} \sum_{e \in G_l^{(t)}} d_j(e_i, e) \tag{23}$$

(2) If the matching function is given by equation (12), assign individual $e_i \in E$ to cluster C_m according to:

$$m = (\chi^{(t)}(e_i))^{(t)} = argmin_{1 \leq r \leq c} \sum_{l=1}^{c} K^T(\delta(r,l)) \sum_{j=1}^{p} (\lambda_{lj}^{(t)})^s \sum_{e \in G_l^{(t)}} d_j(e_i, e) \tag{24}$$

(3) If the matching function is given by equation (13), , assign individual $e_i \in E$ to cluster C_m according to:

$$m = (\chi^{(t)}(e_i))^{(t)} = argmin_{1 \leq r \leq c} \sum_{l=1}^{c} K^T(\delta(r,l)) \sum_{j=1}^{p} \lambda_{j}^{(t)} \sum_{e \in G_l^{(t)}} d_j(e_i, e) \tag{25}$$

(4) If the matching function is given by equation (14), assign individual $e_i \in E$ to cluster C_m according to:

$$m = (\chi^{(t)}(e_i))^{(t)} = argmin_{1 \leq r \leq c} \sum_{l=1}^{c} K^T(\delta(r,l)) \sum_{j=1}^{p} (\lambda_{j}^{(t)})^s \sum_{e \in G_l^{(t)}} d_j(e_i, e) \tag{26}$$

3.4 Algorithm

The batch SOM for relational data based on multiple dissimilarity matrices can be summarized as follows:

(1) Initialization
Fix the number c of clusters;
Fix the cardinality $1 \leq q << n$ of the prototypes G_l $(l = 1, \ldots, c)$;
Fix δ; Fix the kernel fuction K
Fix the number of iterations N_{iter}
Fix T_{min}, T_{max}; Set $T \leftarrow T_{max}$; Set $t \leftarrow 0$;
Randomly select c distinct prototypes $G_l^{(0)} \in E^{(q)}$ $(l = 1, \ldots, c)$;
Set $\boldsymbol{\lambda}_l{}^{(0)} = (1, \ldots, 1)$ $(l = 1, \ldots, c)$;
Set the map $L(c, \mathbf{G}^0)$, where $\mathbf{G}^0 = (G_1^{(0)}, \ldots, G_c^{(0)})$
Assign each object e_i to the closest prototype in order to obtain the partition $P^{(0)} = (P_1^{(0)}, \ldots, P_c^{(0)})$ according to equations (23)–(26)
(2) *Representation step: computation of the best prototypes.*
Set $t = t + 1$;

Compute $T = T_{max}(\frac{T_{min}}{T_{max}})^{\frac{t}{N_{iter}-1}}$

The partition $P^{(t-1)} = (P_1^{(t-1)}, \ldots, P_c^{(t-1)})$ and $\boldsymbol{\lambda}_l{}^{(t-1)}\,(l = 1, \ldots, c)$ are kept fixed.

Compute the prototype $G_l^{(t)} = G^* \in E^{(q)}$ of cluster $P_l^{(t-1)}$ $(l = 1, \ldots, c)$ according to equations (15)–(18)

(3) *Weighting step: computation of the best weights.*

The partition $P^{(t-1)} = (P_1^{(t-1)}, \ldots, P_c^{(t-1)})$ and the prototypes $G_l^{(t)} \in E^{(q)}\,(l = 1, \ldots, c)$ are kept fixed.

Compute the vectors of weights $\boldsymbol{\lambda}_l$ $(l = 1, \ldots, c)$ according to equations (19)–(22)

4) *Affectation step: definition of the best partition.*

The prototypes $G_l^{(t)} \in E^{(q)}\,(l = 1, \ldots, c)$ and $\boldsymbol{\lambda}_l{}^{(t)}\,(l = 1, \ldots, c)$ are kept fixed

$P^{(t)} \leftarrow P^{(t-1)}$

for $i = 1$ to n do

find the cluster $C_{m^*}^{(t)}$ to which e_i belongs

find the winning cluster $C_m^{(t)}$ according to equations (23)–(26)

if $m^* \neq m$

$C_m^{(t)} \leftarrow C_m^{(t)} \cup \{e_i\}$

$C_{m^*}^{(t)} \leftarrow C_m^{(t)} \setminus \{e_i\}$

4) *Stopping criterion.*

If $T = T_{min}$ (or if $t = N_{iter} - 1$) then STOP; otherwise go to 2 (Representation step).

3.5 Properties of the algorithms

This section illustrates the convergence properties of the presented algorithm by giving the proof of the convergence of the $MRDCA - RWL$ clustering algorithm introduced in section **??**. Then, the complexity of both $MRDCA - RWL$ and $MRDCA - RWG$ clustering algorithms are given.

3.5.1 Convergence properties

According to the general schema of the dynamic clustering algorithm [30], this clustering method looks for the partition $P^* = \{C_1^*, \ldots, C_K^*\}$ of E into K clusters, the corresponding K prototypes $\mathbf{G}^* = (G_1^*, \ldots, G_K^*)$ representing the clusters in P^* and K squared adaptive Euclidean distances parameterized by K vectors of weights $\mathbf{D}^* = (\boldsymbol{\lambda}_1^*, \ldots, \boldsymbol{\lambda}_K^*)$ such that

$$W(\mathbf{G}^*, \mathbf{D}^*, P^*) = min\left\{W(\mathbf{G}, \mathbf{D}, P) : \mathbf{G} \in \mathbb{L}^K, \mathbf{D} \in \mathbf{\Lambda}^K, P \in \mathbb{P}_K\right\} \quad (27)$$

where

- $\mathbb{P}_K$ is the set of all the possible partitions of E in K classes such that $C_k \in \mathbf{P}(E)$ (the set of subsets of E) and $P \in \mathbb{P}_K$;
- $\mathbb{L}$ is the representation space of prototypes such that $G_k \in \mathbb{L}\,(k = 1, \ldots, K)$ and $\mathbf{G} \in \mathbb{L}^K = \mathbb{L} \times \ldots \times \mathbb{L}$. In this paper, $\mathbb{L} = E^{(q)} = \{A \subset E : |A| = q\}$.
- $\mathbf{\Lambda}$ is the space of vectors of weights that parameterize the adaptive Euclidean distances such that $\boldsymbol{\lambda}_k \in \mathbf{\Lambda}(k = 1, \ldots, K)$. Here, $\mathbf{\Lambda}$= $\{\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_p) \in \mathbb{R}^p : \lambda_j > 0$ and $\prod_{j=1}^{p} \lambda_j = 1\}$ and $\mathbf{D} \in \mathbf{\Lambda}^K = \mathbf{\Lambda} \times \ldots \times \mathbf{\Lambda}$.

According to [30], the properties of convergence of this kind of algorithm can be studied from two series: $v_t = (\mathbf{G}^t, \mathbf{D}^t, P^t) \in \mathbb{L}^K \times \mathbf{\Lambda}^K \times \mathbb{P}_K$ and $u_t = J(v_t) = J(\mathbf{G}^t, \mathbf{D}^t, P^t),\ t = 0, 1, \ldots$. From an initial term $v_0 = (\mathbf{G}^0, \mathbf{D}^0, P^0)$, the algorithm computes the different terms of the series v_t until the convergence (to be shown) when the criterion J achieves a stationary value.

Proposition 3.4 *The series $u_t = J(v_t)$ decreases at each iteration and converges.*

Proof.

Following [30], first the authors of this study show that the inequalities (I), (II) and (III)

$$\underbrace{J(\mathbf{G}^t, \mathbf{D}^t, P^t)}_{u_t} \overset{(I)}{\geq} J(\mathbf{G}^{t+1}, \mathbf{D}^t, P^t) \overset{(II)}{\geq} J(\mathbf{G}^{t+1}, \mathbf{D}^{t+1}, P^t) \overset{(III)}{\geq} \underbrace{J(\mathbf{G}^{t+1}, \mathbf{D}^{t+1}, P^{t+1})}_{u_{t+1}}$$

hold (i.e., the series decreases at each iteration).

The inequality (I) holds because

$J(\mathbf{G}^t, \mathbf{D}^t, P^t) = \sum_{k=1}^{K} \sum_{e_i \in C_k^{(t)}} D_{\boldsymbol{\lambda}_k^{(t)}}(e_i, G_k^{(t)})$,

$J(\mathbf{G}^{t+1}, \mathbf{D}^t, P^t) = \sum_{k=1}^{K} \sum_{e_i \in C_k^{(t)}} D_{\boldsymbol{\lambda}_k^{(t)}}(e_i, G_k^{(t+1)})$,

and according to proposition (**??**),

$$\mathbf{G}^{(t+1)} = (G_1^{(t+1)}, \ldots, G_K^{(t+1)}) = \underbrace{argmin}_{\mathbf{G}=(G_1,\ldots,G_K)\in\mathbb{L}^K} \sum_{k=1}^{K} \sum_{e_i \in C_k^{(t)}} D_{\boldsymbol{\lambda}_k^{(t)}}(e_i, G_k).$$

Moreover, inequality (II) also holds because

$$J(\mathbf{G}^{t+1}, \mathbf{D}^{(t+1)}, P^t) = \sum_{k=1}^{K} \sum_{e_i \in C_k^{(t)}} D_{\boldsymbol{\lambda}_k^{(t+1)}}(e_i, G_k^{(t+1)}),$$

and according to proposition (**??**),

$$\mathbf{D}^{t+1} = (\boldsymbol{\lambda}_1^{(t+1)}, \ldots, \boldsymbol{\lambda}_K^{(t+1)}) = \underbrace{argmin}_{\mathbf{D}=(\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_K) \in \boldsymbol{\Lambda}^K} \sum_{k=1}^{K} \sum_{e_i \in C_k^{(t)}} D_{\boldsymbol{\lambda}_k}(e_i, G_k^{(t+1)})$$

The inequality (III) also holds because

$$J(\mathbf{G}^{t+1}, \mathbf{D}^{t+1}, P^{t+1}) = \sum_{e_i \in C_k^{(t+1)}} D_{\boldsymbol{\lambda}_k^{(t+1)}}(e_i, G_k^{(t+1)}),$$

and according to proposition (**??**),

$$P^{t+1} = \{C_1^{t+1}, \ldots, C_K^{t+1}\} = \underbrace{argmin}_{P=\{C_1, \ldots, C_K\} \in \mathbb{P}_K} \sum_{k=1}^{K} \sum_{e_i \in C_k} D_{\boldsymbol{\lambda}_k^{(t+1)}}(e_i, G_k^{(t+1)}).$$

Finally, because the series u_t decreases and it is bounded ($J(v_t) \geq 0$), it converges.

Proposition 3.5 *The series $v_t = (\boldsymbol{G}^t, \boldsymbol{D}^t, P^t)$ converges.*

Proof. Assume that the stationarity of the series u_t is achieved in the iteration $t = T$. Then, it is seen that $u_T = u_{T+1}$ and then $J(v_T) = J(v_{T+1})$.

From $J(v_T) = J(v_{T+1})$, one has $J(\mathbf{G}^t, \mathbf{D}^t, P^t) = J(\mathbf{G}^{T+1}, \mathbf{D}^{T+1}, P^{T+1})$ and this equality, according to proposition 3.4, can be rewritten as equalities (I), (II) and (III):

$$J(\mathbf{G}^t, \mathbf{D}^t, P^t) \overset{I}{=} J(\mathbf{G}^{T+1}, \mathbf{D}^T, P^T) \overset{II}{=} J(\mathbf{G}^{T+1}, \mathbf{D}^{T+1}, P^T) \overset{III}{=} J(\mathbf{G}^{T+1}, \mathbf{D}^{T+1}, P^{T+1})$$

3.5.2 Complexity analysis

From the first equality (I), one can understand that $\mathbf{G}^T = \mathbf{G}^{T+1}$ because $\mathbf{G}$ is unique in minimizing J when the partition P^T and the vector of vectors of weights $\mathbf{D}^T$ are fixed. From the second equality (II), $\mathbf{D}^T = \mathbf{D}^{T+1}$ because $\mathbf{D}$ is unique in minimizing J when the partition P^T and the vector of prototypes $\mathbf{G}^{T+1}$ are fixed. Moreover, from the third equality (III), $P^T = P^{T+1}$ because P is unique in minimizing J (because if the minimum is not unique, e_i is assigned to the cluster having the smallest index) when the vector of prototypes $\mathbf{G}^{T+1}$ and the vector vectors of weights $\mathbf{D}^T$ are fixed.

Finally, one can conclude that $v_T = v_{T+1}$. This conclusion holds for all $t \geq T$ and $v_t = v_T, \forall t \geq T$, and it follows that the series v_t converges.

The time complexity of $MRDCA - RWL$ can be analyzed considering the complexity of each single step. Let n be the number of objects, $K << n$ be the number of clusters, $q << n$ be the cardinality of each prototype and p be the number of dissimilarity matrices.

- *Initialization.* In this step, the initialization of the relevance weight vector costs $O(K \times p)$. The random selection of K distinct prototypes (i.e., the selection of $K \times q$ distinct objects) can be done using random functions and a red-black tree to check for repetitions. The time complexity is then $O(K \times q \times log(K \times q))$. The assignment of each object to the closest prototype corresponds to the step 3. The complexity is $O(n \times K \times q \times p)$. Thus, the initialization costs $O(n \times K \times q \times p)$.
- *Step1: computation of the best prototypes.* For each cluster using each table of dissimilarity the authors test each individual as a candidate prototype. This needs the computation of the distance between an individual $i\,(i = 1, ...n)$ and all elements of each cluster using all p dissimilarity matrices and it costs $O(n^2 * p)$. The selection of the prototype of cardinality q for each cluster needs to sort the vector of individual for each cluster (it costs $O(K \times n \times log\, n)$) and to select the best q individuals as the prototype (it costs $O(K \times q)$). Thus, the step 1 costs $O(n^2 * p)$.
- *Step 2: computation of the best relevance weight vectors.* According to equation (**??**), this step needs the computation of K denominators, the computation of the numerator just once, and to repeat that for each component of the vector of relevance weights. Thus, the step 2 costs $O(n \times q \times p + K \times p)$.
- *Step 3: definition of the best partition.* This step needs the computation of the dissimilarity between an individual $i\,(i = 1, \ldots, n)$ and the prototype of cardinality q of each cluster using the p dissimilarity matrices and it costs $O(n \times q \times K \times p)$.

So, globally these steps cost $O(n^2 * p)$. Thus, if the clustering process needs t iterations to converge, the total time complexity of this algorithm is $O(n^2 \times p \times t)$. Following a similar reasoning, one can conclude that the total time complexity of $MRDCA - RWG$ is also $O(n^2 \times p \times t)$.

4 Empirical results

To evaluate the performance of these partitioning relational hard clustering algorithms in comparison with $NERF$ and $SRDCA$ (relational clustering algorithms that perform on a single dissimilarity matrix) as well as $MRDCA$ (relational hard clustering algorithm that performs on multiple dissimilarity

matrices) and $CARD-R$ (relational fuzzy clustering algorithm that performs on multiple dissimilarity matrices and learns a relevance weight for each dissimilarity matrix in each cluster), applications with synthetic and real data sets (available at the UCI Repository http://www.ics.uci.edu/mlearn/ ML-Repository.html) described by real-valued variables as well as time trajectories data sets (available at http://www.math.univ-toulouse.fr/staph/npfda/ npfda-datasets.html) are considered.

The relational hard clustering algorithms $SRDCA$, $MRDCA$, $MRDCA - RWL$ and $MRDCA - RWG$ will be applied to these data sets to obtain a partition $Q = (Q_1, ..., Q_K)$. $NERF$ and $CARD - R$ will be also applied to these data sets to obtain first a fuzzy partition into K fuzzy clusters. Then, a hard partition $Q = (Q_1, \ldots, Q_K)$ is obtained from this fuzzy partition by defining the hard cluster $Q_k (k = 1, \ldots, K)$ as: $Q_k = \{e_i : u_{ik} \geq u_{im} \, \forall m \in \{1, \ldots, K\}\}$. The quantity u_{ik} is the membership degree of object $e_i \, (i = 1, \ldots, n)$ in fuzzy cluster $k \, (k = 1, \ldots, K)$.

To compare the clustering results furnished by the clustering methods, an external index – the corrected Rand index (CR) [31] – as well as the $F - measure$ [32] and the overall error rate of classification ($OERC$) [33] will be considered.

Let $P = \{P_1, \ldots, P_i, \ldots, P_m\}$ be the *a priori* partition into m classes and $Q = \{Q_1, \ldots, Q_j, \ldots, Q_K\}$ be the hard partition into K clusters given by a clustering algorithm. Let the confusion matrix be as below:

Table 1
Confusion matrix

	Clusters					
Classes	Q_1	$\ldots$	Q_j	$\ldots$	Q_K	$\sum$
P_1	n_{11}	$\ldots$	n_{1j}	$\ldots$	n_{1K}	$n_{1\bullet} = \sum_{j=1}^{K} n_{1j}$
$\vdots$	$\vdots$	$\ldots$	$\vdots$	$\ldots$	$\vdots$	$\vdots$
P_i	n_{i1}	$\ldots$	n_{ij}	$\ldots$	n_{iK}	$n_{i\bullet} = \sum_{j=1}^{K} n_{ij}$
$\vdots$	$\vdots$	$\ldots$	$\vdots$	$\ldots$	$\vdots$	
P_m	n_{m1}	$\ldots$	n_{mj}	$\ldots$	n_{mK}	$n_{m\bullet} = \sum_{j=1}^{K} n_{mj}$
$\sum$	$n_{\bullet 1} = \sum_{i=1}^{m} n_{i1}$	$\ldots$	$n_{\bullet j} = \sum_{i=1}^{m} n_{ij}$	$\ldots$	$n_{\bullet K} = \sum_{i=1}^{m} n_{iK}$	$n = \sum_{i=1}^{m} \sum_{j=1}^{K} n_{ij}$

The corrected Rand index is:

$$CR = \frac{\sum_{i=1}^{m} \sum_{j=1}^{K} \binom{n_{ij}}{2} - \binom{n}{2}^{-1} \sum_{i=1}^{m} \binom{n_{i\bullet}}{2} \sum_{j=1}^{K} \binom{n_{\bullet j}}{2}}{\frac{1}{2}[\sum_{i=1}^{m} \binom{n_{i\bullet}}{2} + \sum_{j=1}^{K} \binom{n_{\bullet j}}{2}] - \binom{n}{2}^{-1} \sum_{i=1}^{m} \binom{n_{i\bullet}}{2} \sum_{j=1}^{K} \binom{n_{\bullet j}}{2}} \tag{28}$$

where $\binom{n}{2} = \frac{n(n-1)}{2}$ and n_{ij} represents the number of objects that are in class P_i and cluster Q_j; $n_{i\bullet}$ indicates the number of objects in class P_i; $n_{\bullet j}$ indicates

the number of objects in cluster Q_j; and n is the total number of objects in the data set.

CR index assesses the degree of agreement (similarity) between an a priori partition and a partition furnished by the clustering algorithm. Moreover, the CR index is not sensitive to the number of classes in the partitions or the distribution of the items in the clusters. Finally, CR index takes its values from the interval [-1,1], in which the value 1 indicates perfect agreement between partitions, whereas values near 0 (or negatives) correspond to cluster agreement found by chance [34].

The traditional $F-measure$ between class $P_i\,(i=1,\ldots,m)$ and cluster $Q_j\,(j=1,\ldots,K)$ is the harmonic mean of precision and recall:

$$F-measure(P_i,Q_j)=2\,\frac{Precision(P_i,Q_j)\,Recall(P_i,Q_j)}{Precision(P_i,Q_j)+Recall(P_i,Q_j)} \tag{29}$$

The $Precision$ between class $P_i\,(i=1,\ldots,m)$ and cluster $Q_j\,(j=1,\ldots,K)$ is defined as the ratio between the number of objects that are in class P_i and cluster Q_j and the number of objects in cluster Q_j:

$$Precision(P_i,Q_j)=\frac{n_{ij}}{n_{\bullet j}}=\frac{n_{ij}}{\sum_{i=1}^{m}n_{ij}} \tag{30}$$

The $Recall$ between class $P_i\,(i=1,\ldots,m)$ and cluster $Q_j\,(j=1,\ldots,K)$ is defined as the ratio between the number of objects that are in class P_i and cluster Q_j and the number of objects in class P_i:

$$Recall(P_i,Q_j)=\frac{n_{ij}}{n_{i\bullet}}=\frac{n_{ij}}{\sum_{j=1}^{K}n_{ij}} \tag{31}$$

The $F-measure$ between the *a priori* partition $P=\{P_1,\ldots,P_i,\ldots,P_m\}$ and the hard partition $Q=\{Q_1,\ldots,Q_j,\ldots,Q_K\}$ given by a cluster algorithm is defined as:

$$F-measure(P,Q)=\frac{1}{n}\sum_{i=1}^{m}n_{i\bullet}\ \max_{1\le j\le K}\ F-measure(P_i,Q_j) \tag{32}$$

The $F-measure$ index takes its values from the interval [0,1], in which the value 1 indicates perfect agreement between partitions.

In classification problems, each cluster Q_j is assigned to an *a priori* class P_i and this assignment must be interpreted as if the true *a priori* class is P_i. Once this decision is taken, for a given object of the cluster Q_j the decision is correct if the *a priori* class of this object is P_i and is an error if the *a priori* class is not P_i. To have a minimum error rate of classification ERC, one needs to seek a decision rule that minimizes the probability of error.

Let $p(P_i/Q_j)$ be the posterior probability that an object belongs to the class P_i when it is assigned to the cluster Q_j. Let $p(Q_j)$ the probability that the object belongs to the cluster Q_j. The function p is known as the likelihood function.

The maximum a posteriori probability (MAP) estimate is the mode of the posteriori probability $p(P_i/Q_j)$ and the index of the *a priori* class associated to this mode is given by:

$$MAP(Q_j) = \arg \max_{1 \leq i \leq m} p(P_i/Q_j) \tag{33}$$

The Bayes decision rule to minimize the average probability of error is to select the *a priori* class that maximizes the posterior probability. The error rate of classification $ERC(Q_j)$ of the cluster Q_j is equal to $1 - p(P_{MAP(Q_j)}/Q_j)$ and the overall error rate of classification $OERC$ is equal to :

$$OERC = \sum_{j=1}^{K} p(Q_j)(1 - p(P_{MAP(Q_j)}/Q_j)) \tag{34}$$

For a sample,

$$p(P_{MAP(Q_j)}/Q_j) = \max_{1 \leq i \leq m} \frac{n_{ij}}{n_{\bullet j}}. \tag{35}$$

The $OERC$ index aims to measure the ability of a clustering algorithm to find out the *a priori* classes present in a data set and it is computed by :

$$OERC = \sum_{j=1}^{K} \frac{n_{\bullet j}}{n} (1 - \max_{1 \leq i \leq m} n_{ij}/n_{\bullet j}) = 1 - \frac{\sum_{j=1}^{K} \max_{1 \leq i \leq m} n_{ij}}{n} \tag{36}$$

4.1 Synthetic real valued data sets

This paper considers data sets described by two real-valued variables. Each data set has 450 points scattered among four classes of unequal sizes and elliptical shapes: two classes of size 150 each and two classes of sizes 50 and 100.

Each class in these quantitative data sets was drawn according to a bivariate normal distribution.

Four different configurations of real-valued data drawn from bivariate normal distributions according to each class are considered. These distributions have the same mean vector (see Table 2), but different covariance matrices (see Table 3): 1) the variance are different between the variables and from one class to another (synthetic data set 1); 2) the variance are different between the variables but they are almost the same from one class to another (synthetic data set 2); 3) the variance are almost the same between the variables and different from one class to another (synthetic data set 3). 4) the variance are almost the same between the variables and from one class to another (synthetic data set 3).

Table 2
Configurations of quantitative data sets: mean vectors of the bivariate normal distributions of the classes.

$\boldsymbol{\mu}$	Class 1	Class 2	Class 3	Class 4
μ_1	45	70	45	42
μ_2	30	38	35	20

Table 3
Configurations of quantitative data sets: covariance matrices of the bivariate normal distributions of the classes.

$\boldsymbol{\Sigma}$	Synthetic data set 1				Synthetic data set 2			
	Class 1	Class 2	Class 3	Class 4	Class 1	Class 2	Class 3	Class 4
σ_1	100	20	50	1	15	15	15	15
σ_2	1	70	40	10	5	5	5	5
ρ_{12}	0.88	0.87	0.90	0.89	0.88	0.87	0.90	0.89
$\boldsymbol{\Sigma}$	Synthetic data set 3				Synthetic data set 4			
	Class 1	Class 2	Class 3	Class 4	Class 1	Class 2	Class 3	Class 4
σ_1	16	10	2	6	8	8	8	8
σ_2	15	11	1	5	7	7	7	7
ρ_{12}	0.78	0.77	0.773	0.777	0.78	0.77	0.773	0.777

Several dissimilarity matrices are obtained from these data sets. One of these dissimilarity matrices has the cells that are the dissimilarities between pairs of objects computed taking into account simultaneously the two real-valued attributes. All the others dissimilarity matrices have the cells that are the dissimilarities between pairs of objects computed taking into account only a single real-valued attribute. Because all the attributes are real-valued, distance functions belonging to the family of Minkowsky distance (Manhattan

or "city-block" distance, Euclidean distance, Chebyshev distance, etc.) are suitable to compute dissimilarities between the objects. In this paper, the dissimilarity between pairs of objects were computed according to the Euclidean (L_2) distance.

For these data sets, $NERF$ and $SRDCA$ are performed on the dissimilarity matrix that has the cells that are the dissimilarities between pairs of objects computed taking into account simultaneously the two real-valued attributes. $CARD-R$, $MRDCA$, $MRDCA-RWL$ and $MRDCA-RWG$ are performed simultaneously on all dissimilarity matrices which have the cells that are the dissimilarities between pairs of objects computed taking into account only a single real-valued attribute.

All dissimilarity matrices were normalized according to their overall dispersion [37] to have the same dynamic range. This means that each dissimilarity $d(e_k, e_{k'})$ in a given dissimilarity matrix has been normalized as $\frac{d(e_k,e_{k'})}{T}$ where $T = \sum_{k=1}^{n} d(e_k, g)$ is the overall dispersion and $g = e_l \in E = \{e_1, \ldots, e_n\}$is the overall prototype, which is computed according to $l = argmin_{1 \leq h \leq n} \sum_{k=1}^{n} d(e_k, e_h)$.

The relational fuzzy clustering algorithms $NERF$ and $CARD-R$ were applied to the dissimilarity matrices obtained from this data set to obtain a four-cluster fuzzy partition. The hard clustering algorithms $SRDCA$, $MRDCA$, $MRDCA-RWL$ and $MRDCA-RWG$ were applied to the dissimilarity matrices obtained from this data set to obtain a four-cluster hard partition. The hard cluster partitions (obtained from the fuzzy partitions given by $NERF$ or $CARD-R$ or obtained directly from $SRDCA$, $MRDCA$, $MRDCA-RWL$ and $MRDCA-RWG$) were compared with the known a priori class partition. For the synthetic data sets, the CR, $F-measure$ and $OERC$ indexes were estimated in the framework of a Monte Carlo simulation with 100 replications. The average and the standard deviation of these indexes between these 100 replications were calculated. In each replication, a relational clustering algorithm was run (until the convergence to a stationary value of the adequacy criterion) 100 times and the best result was selected according to the adequacy criterion. The CR, $F-measure$ and $OERC$ indexes were calculated for the best result.

Table 4 shows the performance of the $NERF$ and the $CARD-R$ algorithms as well as the performance of $SRDCA$, $MRDCA$, $MRDCA-RWL$ and $MRDCA-RWG$ algorithms (with prototypes of cardinality $|G_k| = 1$, $k = 1, \ldots, 4$) on the synthetic data sets according to the average and the standard deviation of the CR, $F-measure$ and $OERC$ indexes. Table 5 shows the 95% confidence interval for the average of the CR, $F-measure$ and $OERC$ indexes.

The performance of the algorithms $MRDCA-RWL$ and $CARD-R$ was clearly superior when the variance was different between the variables and from one class to another (synthetic data set 1), in comparison with all the other algorithms. $MRDCA-RWL$, $CARD-R$ and $MRDCA-RWG$ algorithms were also superior when the variance was different between the variables but almost the same from one class to another (synthetic data set 2) especially in comparison with the algorithms that perform on a single dissimilarity matrix ($NERF$ and $SRDCA$). Moreover, except $CARD-R$, which presented its worst performance, all the other algorithms had a similar performance when the variance was almost the same between the variables and different from one class to another (synthetic data set 3). Finally, the algorithms $NERF$ and $SRDCA$ were superior in comparison with all the other algorithms when the variance was almost the same between the variables and from one class to another (synthetic data set 4). In conclusion, in comparison with the algorithms that perform on a single dissimilarity matrix, $MRDCA-RWL$ was clearly superior in the synthetic data sets where the variance was different between the variables whereas $MRDCA-RWG$ was clearly superior only in the synthetic data sets where the variance was different between the variables but almost the same from one class to another.

4.2 UCI machine learning repository data sets

This paper considers abalone, image, iris plants, thyroid gland, and wine data sets. These data sets are found in http://www.ics.uci.edu/mlearn/ MLRepository.html.

All these data sets are described by a data matrix of "objects $\times$ real-valued attributes". Several dissimilarity matrices are obtained from these data matrices. One of these dissimilarity matrices has the cells that are the dissimilarities between pairs of objects computed taking into account simultaneously all the real-valued attributes. All the other dissimilarity matrices have the cells which are the dissimilarities between pairs of objects computed taking into account only a single real-valued attribute. In this paper, the dissimilarity between pairs of objects were computed according to the Euclidean (L_2) distance.

For these data sets, $NERF$ and $SRDCA$ were performed on the dissimilarity matrix which has the cells that are the dissimilarities between pairs of objects computed taking into account simultaneously all the real-valued attributes. $CARD-R$, $MRDCA$, $MRDCA-RWL$ and $MRDCA-RWG$ were performed simultaneously on all dissimilarity matrices which have the cells that are the dissimilarities between pairs of objects computed taking into account only a single real-valued attribute. All dissimilarity matrices were normalized according to their overall dispersion [37] to have the same dynamic range.

Table 4
Performance of the algorithms on the synthetic data sets: average and standard deviation (in parenthesis) of the CR, $F-measure$, and $OERC$ indexes

Algorithms	Synthetic data set 1		
	CR	$F-measure$	$OERC$
$NERF$	0.1334 (0.0206)	0.4942 (0.0187)	42.98% (1.88%)
$SRDCA$	0.1118 (0.0221)	0.4861 (0.0207)	44.57% (2,19%)
$MRDCA$	0.1008 (0.0236)	0.4645 (0.0257)	46.14% (2.50%)
$MRDCA-RWG$	0.1137 (0.0250)	0.4790 (0.0239)	44.98% (2.44%)
$MRDCA-RWL$	0.5327 (0.0281)	0.7080 (0.0262)	23.76% (2.54%)
$CARD-R$	0.4810 (0.0296)	0.6947 (0.0217)	25.69% (2.35%)
Algorithms	Synthetic data set 2		
	CR	$F-measure$	$OERC$
$NERF$	0.1416 (0.0173)	0.4730 (0.0186)	46.26% (1.72%)
$SRDCA$	0.1415 (0.0178)	0.4728 (0.0212)	46.13% (1.79%)
$MRDCA$	0.2066 (0.0264)	0.5486 (0.0291)	39.82% (2.87%)
$MRDCA-RWG$	0.2384 (0.0327)	0.5763 (0.0309)	37.54% (2.90%)
$MRDCA-RWL$	0.2343 (0.0414)	0.5672 (0.0423)	38.26% (3.96%)
$CARD-R$	0.2571 (0.0214)	0.5828 (0.0212)	36.88% (1.89%)
Algorithms	Synthetic data set 3		
	CR	$F-measure$	$OERC$
$NERF$	0.2381 (0.0279)	0.5294 (0.0244)	41.75% (2.41%)
$SRDCA$	0.2172 (0.0446)	0.5189 (0.0383)	43.20% (3.36%)
$MRDCA$	0.2353 (0.0439)	0.5448 (0.0284)	41.93% (3.20%)
$MRDCA-RWG$	0.2133 (0.0368)	0.5123 (0.0312)	43.07% (3.15%)
$MRDCA-RWL$	0.2208 (0.0296)	0,5180 (0.0301)	43.44% (2.66%)
$CARD-R$	0.1285 (0.0130)	0.4395 (0.0200)	51.73% (1.76%)
Algorithms	Synthetic data set 4		
	CR	$F-measure$	$OERC$
$NERF$	0.2942 (0.0285)	0.6013 (0.0267)	34.79% (2.48%)
$SRDCA$	0.3014 (0.0307)	0.6034 (0.0214)	34.54% (2.49%)
$MRDCA$	0.2741 (0.0312)	0,5910 (0.0250)	35.83% (2.79%)
$MRDCA-RWG$	0.2888 (0.0276)	0.5885 (0.0244)	35.55% (2.08%)
$MRDCA-RWL$	0.2873 (0.0313)	0.5826 (0.0316)	35.96% (2.91%)
$CARD-R$	0.1625 (0.0190)	0.5026 (0.0207)	43.56% (2.24%)

Each relational clustering algorithm was run (until the convergence to a stationary value of the adequacy criterion) 100 times and the best result was selected according to the adequacy criterion. The hard cluster partitions (obtained from the fuzzy partitions given by $NERF$ or $CARD-R$ or obtained directly from $SRDCA$, $MRDCA$, $MRDCA-RWL$ and $MRDCA-RWG$) were compared with the known a priori class partition. The comparison criteria used were the corrected Rand index (CR), the $F-measure$ and the overall error rate of classification ($OERC$). The CR, $F-measure$, and $OERC$ indexes were calculated for the best result.

Table 5
Performance of the algorithms on the synthetic data sets: 95% confidence interval for the average of the CR, $F-measure$, and $OERC$ indexes

Algorithms	Synthetic data set 1		
	CR	$F-measure$	$OERC$
$NERF$	0.1293—0.1374	0.4905—0.4978	42.61%—43.35%
$SRDCA$	0.1074—0.1161	0.4820—0.4901	44.14%—45.00%
$MRDCA$	0.0961—0.1054	0.4594—0.4695	45.65%—46.63%
$MRDCA-RWG$	0.1088—0.1186	0.4743—0.4836	44,50%—45.45%
$MRDCA-RWL$	0.5271—0.5382	0.7028—0.7131	23.26%—24.25%
$CARD-R$	0.4751—0.4868	0.6904—0.6989	25.23%—26.15%
Algorithms	Synthetic data set 2		
	CR	$F-measure$	$OERC$
$NERF$	0.1382—0.1449	0.4693—0.4766	45.92%—46.60%
$SRDCA$	0.1380—0.1449	0.4686—0.4769	45.78%—46.48%
$MRDCA$	0.2014—0.2117	0.5428—0.5543	39.26%—40.39%
$MRDCA-RWG$	0.2319—0.2448	0.5702—0.5823	36.97%—38.11%
$MRDCA-RWL$	0.2261—0.2424	0.5589—0.5754	37.48%—39.04%
$CARD-R$	0.2529—0.2612	0.5786—0.5869	36.51%—37.25%
Algorithms	Synthetic data set 3		
	CR	$F-measure$	$OERC$
$NERF$	0.2326—0.2435	0.5246—0.5341	41.27%—42.22%
$SRDCA$	0.2084—0.2259	0.5113—0.5264	42.54%—43.86%
$MRDCA$	0.2266—0.2439	0.5392—0.5503	41.30%—42.56%
$MRDCA-RWG$	0.2060—0.2205	0.5061—0.5184	42.45%—43.69%
$MRDCA-RWL$	0.2149—0.2266	0.5121—0.5238	42.92%—43.96%
$CARD-R$	0.1259—0.1310	0.4355—0.4434	51.38%—52.07%
Algorithms	Synthetic data set 4		
	CR	$F-measure$	$OERC$
$NERF$	0.2886—0.2997	0.5960—0.6065	34.30%—35.27%
$SRDCA$	0.2953—0.3074	0.5992—0.6075	34.05%—35.03%
$MRDCA$	0.2679—0.2802	0.5861—0.5959	35.28%—36.38%
$MRDCA-RWG$	0.2833—0.2942	0.5837—0.5932	35.14%—35.96%
$MRDCA-RWL$	0.2811—0.2934	0.5764—0.5887	35.38%—36.53%
$CARD-R$	0.1587—0.1662	0.4985—0.5066	43.12%—44.00%

4.2.1 Iris data set

This data set consists of three types (classes) of iris plants: iris setosa, iris versicolour and iris virginica. The three classes each have 50 instances (objects). One class is linearly separable from the other two; the latter two are not linearly separable from each other. Each object is described by four real-valued attributes: (1) sepal length, (2) sepal width, (3) petal length and (4) petal width.

Table 8 gives the final relevance weight matrix. In this table, the weights for

Table 6
Iris data set: confusion matrix from B-SOM algorithm ($T_{max} = 6$)

	Classes			
Clusters	1-Iris setosa	2-Iris versicolour	3-Iris virginica	Majority Class
0,0	0	0	0	-
0,1	0	0	3	3
0,2	0	11	0	2
0,3	0	12	0	2
0,4	0	10	0	2
0,5	0	14	1	2
0,6	7	0	0	1
0,7	22	0	0	1
1,0	0	0	0	-
1,1	0	0	8	3
1,2	0	0	10	3
1,3	0	0	14	3
1,4	0	2	9	3
1,5	0	1	5	3
1,6	0	0	0	-
1,7	21	0	0	1

Table 7
Iris data set: confusion matrix from AB-SOM algorithm for multiple dissimilarity data tables ($T_{max} = 6$)

	Classes			
Clusters	1-Iris setosa	2-Iris versicolour	3-Iris virginica	Majority Class
0,0	0	16	0	2
0,1	0	12	0	2
0,2	0	1	10	3
0,3	0	4	0	2
0,4	0	0	12	3
0,5	0	0	3	3
0,6	0	0	0	-
0,7	38	0	0	1
1,0	0	2	10	3
1,1	0	9	0	2
1,2	0	3	0	2
1,3	0	0	6	3
1,4	0	0	4	3
1,5	0	3	3	2/3
1,6	0	0	2	3
1,7	12	0	0	1

the matrices 3 and 4 have greater values then the other matrices, denoting a greater influence on the objects affectation to the clusters.

Table 8
Iris: Final relevance weight matrix

Cluster/Matrix	1	2	3	4
0,0	0.63	0.18	**3.40**	**2.62**
0,1	0.80	0.81	**1.45**	**1.04**
0,2	0.34	0.26	**2.95**	**3.79**
0,3	0.14	0.29	**10.14**	**2.26**
0,4	0.58	0.32	**4.99**	**1.05**
1,0	0.25	0.07	**5.06**	**11.38**
1,1	**1.59**	0.52	0.95	**1.26**
1,2	0.35	0.50	**1.50**	**3.73**
1,3	0.25	**1.53**	**1.47**	**1.76**
1,4	0.51	0.26	**2.59**	**2.83**

Table 9
Iris data set: Final relevance weight matrix from AB-SOM algorithm for multiple dissimilarity data tables ($T_{max} = 6$)

	Matrix			
Clusters	1-Sepal length	2-Sepal width	3-Petal length	4-Petal width
0,0	0.5878	0.5075	**2.1617**	**1.5507**
0,1	0.1438	0.7925	**1.5381**	**5.7034**
0,2	0.1193	**47.9332**	0.5631	0.3104
0,3	0.7568	0.9139	**1.7698**	0.8169
0,4	0.3581	0.3508	**3.3944**	**2.3451**
0,5	**1.7980**	0.2266	**2.7677**	0.8866
0,6	**1.0037**	0.3404	**1.6919**	**1.7297**
0,7	0.2591	0.0672	**5.2699**	**10.8944**
1,0	0.4403	0.3797	**4.1288**	**1.4484**
1,1	0.2676	0.2180	**4.7063**	**3.6428**
1,2	0.1780	**2.1141**	**1.1214**	**2.3694**
1,3	**1.7573**	0.0686	**3.2489**	**2.5531**
1,4	0.3215	0.1939	**4.4941**	**3.5684**
1,5	**1.4477**	0.1896	**1.0308**	**3.5350**
1,6	**2.0289**	0.7747	0.6483	0.9812
1,7	0.4077	0.1109	**5.3766**	**4.1132**

4.2.2 Abalone data set

This data set consists of 4177 abalones described by 8 real-valued attributes and 1 nominal attribute. In this application, the 8 real-valued attributes were considered for clustering purposes. They are: (1) Length, (2) Diameter, (3) Height, (4) Whole weight, (5) Shucked weight, (6) Viscera weight, (7) Shell weight and (8) Rings. The nominal attribute "Sex" with three classes (Male, Female and Infant) was used as an *a priori* classification. The classes (Male,

Female and Infant) have 1528, 1307 and 1342 instances, respectively.

The fuzzy clustering algorithms $NERF$ and $CARD-R$ were applied to the dissimilarity matrices obtained from this data set to obtain a three-cluster fuzzy partition. The three-cluster hard partitions obtained from the fuzzy partition were compared with the known a priori three-class partition. $NERF$ had 0.0851, 0.4566 and 51.98% for CR, $F-measure$, and $OERC$ indexes, respectively, whereas $CARD-R$ had 0.0935, 0.5021 and 52.09%, respectively, for these indexes.

The hard clustering algorithms were applied to the dissimilarity matrices obtained from this data set to obtain a three-cluster hard partition. Table 10 shows the performance of the $SRDCA$, $MRDCA$, $MRDCA-RWL$ and $MRDCA-RWG$ algorithms on the abalone data set according to CR, $F-measure$ and $OERC$ indexes, considering prototypes of cardinality $|G_k|$ =1, 2, 3, 5 and 10 ($k=1,2,3$).

For this data set, globally, the best performance was presented by $MRDCA-RWG$, $MRDCA-RWL$, $MRDCA$ and $CARD-R$, in this order. The worst performance was presented by $NERF$ and $SRDCA$, in this order. Note that the performance was stable for $SRDCA$ and worsened for $MRDCA$, $MRDCA-RWL$ and $MRDCA-RWG$, with the increase of the cardinality of the prototypes.

Table 11 gives the vector of relevance weights, globally for all dissimilarity matrices (according to the best result given by $MRDCA-RWG$ algorithm with prototypes of cardinality 1) and locally for each cluster and dissimilarity matrix (according to the best result given by $MRDCA-RWL$ algorithm with prototypes of cardinality 1). Table 12 gives the confusion matrix of the three-cluster hard partition given by the $MRDCA-RWL$ algorithm with prototypes of cardinality 1.

Concerning the three-cluster partition given by $MRDCA-RWG$, dissimilarity matrices were computed taking into account only " (4) Whole weight" and "(5) Shucked weight" attributes which had the highest (5.8800) and the lowest (0.3895) relevance weight in the definition of the clusters, respectively.

For the three-cluster hard partition given by the $MRDCA-RWL$ algorithm, Table 11 shows (in bold) the dissimilarity matrices of the most relevance weights in the definition of each cluster. For example, dissimilarity matrices computed taking into account only "(5) Shucked weight," "(1) Length" and "(2) Diameter" (in this order) are the most relevant in the definition of cluster 3 (Infant).

Table 10
Abalone data set: CR, $F-measure$, and $OERC$ indexes

Indexes	$\lvert G_k\rvert$	$SRDCA$	$MRDCA$	$MRDCA-RWL$	$MRDCA-RWG$
CR	1	0.0855	0.1440	0.1555	0.1847
	2	0.0853	0.1438	0.1531	0.1809
	3	0.0855	0.1436	0.1531	0.1827
	5	0.0855	0.1419	0.1535	0.1809
	10	0.0855	0.1409	0.1531	0.1799
$F-measure$	1	0.4572	0.5398	0.5503	0.6025
	2	0.4570	0.5416	0.5500	0.6005
	3	0.4572	0.5422	0.5500	0.6018
	5	0.4572	0.5402	0.5502	0.6005
	10	0.4572	0.5397	0.5498	0.6010
$OERC$	1	51.92%	46.89%	46.58%	47.11%
	2	51.92%	46.82%	46.66%	47.28%
	3	51.92%	46.89%	46.66%	47.16%
	5	51.92%	47.04%	46.66%	47.33%
	10	51.92%	47.11%	46.68%	47.28%

Table 11
Abalone data set: vectors of relevance weights

Data Matrix	$MRDCA-RWG$	$MRDCA-RWL$		
		Cluster 1	Cluster 2	Cluster 3
Length	**1.0915**	0.2281	**5.8765**	**1.7714**
Diameter	**1.1227**	0.2361	**5.4532**	**1.6567**
Height	0.7615	0.1584	**1.6357**	0.7500
Whole weight	**5.8800**	**84.1655**	0.1887	0.2504
Shucked weight	0.3895	**12.7859**	0.1137	**13.6994**
Viscera weight	0.9774	0.7795	1.1872	0.8345
Shell weight	0.9745	0.6773	0.9422	0.8894
Rings	0.4910	0.2061	0.7943	0.1783

Table 12
Abalone data set: confusion matrix

	Classes		
Clusters	1-Male	2-Female	3-Infant
1	372	256	1068
2	339	346	24
3	817	705	250

4.2.3 Image data set

This data set consists of images that were drawn randomly from a database of seven outdoor images. The images were segmented by hand to create the seven class labels: sky, cement, window, brick, grass, foliage and path. Each class has 330 instances. Each object is described by 16 real-valued attributes. These attributes are: (1) region-centroidcol; (2) region-centroid-row; (3) vedge-mean; (4) vegde-sd; (5) hedge-mean; (6) hedge-sd; (7) intensity-mean; (8) rawred-mean; (9) rawblue-mean; (10) rawgreen-mean; (11) exred-mean; (12) exblue-mean; (13) exgreen-mean; (14) value-mean; (15) saturation-mean and (16) hue-mean.

The fuzzy clustering algorithms $NERF$ and $CARD-R$ were applied to the dissimilarity matrices obtained from this data set to obtain a seven-cluster fuzzy partition. The seven-cluster hard partitions obtained from the fuzzy partition were compared with the known a priori seven-class partition. $NERF$ had 0.2822, 0.5014 and 47.09% for CR, $F-measure$, and $OERC$ indexes, respectively, whereas $CARD-R$ had 0.0528, 0.3051, and 71.47% for these indexes, respectively.

The hard clustering algorithms were applied to the dissimilarity matrices obtained from this data set to obtain a seven-cluster hard partition. Table 13 shows the performance of the $SRDCA$, $MRDCA$, $MRDCA-RWL$ and $MRDCA-RWG$ algorithms on the image data set according to CR, $F-measure$ and $OERC$ indexes, considering prototypes of cardinality $|G_k|$ =1, 2, 3, 5 and 10 ($k=1,\ldots,7$).

For this data set, globally, the best performance was presented by $MRDCA-RWL$, $MRDCA$, $MRDCA-RWG$ and $SRDCA$, in this order. The worst performance was presented by $CARD-R$ and $NERF$, in this order. In particular, $MDCA-RWL$ with prototypes of cardinality 3 had the best and $CARD-R$ had the worst performance, concerning these indexes. Note that the performance was improved for $SRDCA$ and worsened for $MRDCA$, $MRDCA-RWL$ and $MRDCA-RWG$, with the increase of the cardinality of the prototypes.

Table 14 gives the vector of relevance weights, globally for all dissimilarity matrices (according to the best result given by $MRDCA-RWG$ algorithm with prototypes of cardinality 3) and locally for each cluster and dissimilarity matrix (according to the best result given by $MRDCA-RWL$ algorithm with prototypes of cardinality 3). Table 15 gives the confusion matrix of the seven-cluster hard partition given by the $MRDCA-RWL$ algorithm with prototypes of cardinality 3.

Concerning the seven-cluster partition given by $MRDCA-RWG$, dissimilarity matrices computed taking into account only " (16) hue-mean" and

Table 13
Image data set: CR, $F-measure$, and $OERC$ indexes

Indexes	$\lvert G_k\rvert$	$SRDCA$	$MRDCA$	$MRDCA-RWL$	$MRDCA-RWG$
CR	1	0.3116	0.4756	0.4962	0.4382
	2	0.3909	0.4698	0.4947	0.4397
	3	0.3919	0.4603	0.4974	0.4371
	5	0.3223	0.4587	0.4948	0.4123
	10	0.4100	0.4568	0.4949	0.4128
$F-measure$	1	0.5310	0.6342	0.6490	0.6187
	2	0.6116	0.6300	0.6496	0.6101
	3	0.5869	0.6253	0.6527	0.6097
	5	0.5469	0.6237	0.6533	0.5817
	10	0.6193	0.6225	0.6528	0.5841
$OERC$	1	49.48%	38.70%	38.00%	40.12%
	2	44.80%	38.96%	38.05%	37.09%
	3	44.80%	39.61%	37.96%	37.14%
	5	50.04%	39.61%	38.81%	39.69%
	10	41.21%	39.61%	38.26%	39.69%

Table 14
Image data set: vectors of relevance weights

Data Matrix	$MRDCA-RWG$	$MRDCA-RWL$						
		Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	Cluster 6	Cluster 7
region-centroidcol	0.1392	0.0329	0.0433	0.5251	0.0449	0.0764	0.0381	0.1138
region-centroid-row	0.2895	0.1435	0.0309	**5.9588**	0.2166	0.5125	0.1667	0.1974
vedge-mean	0.2570	0.4840	0.1364	0.0867	**2.9086**	0.0699	0.4695	0.0756
vegde-sd	0.6346	**22.1196**	**48.4827**	0.0124	**565.5961**	0.0847	**135.0916**	**15.1192**
hedge-mean	0.2146	0.3846	0.1625	0.0289	0.9601	0.0884	**1.0664**	0.0424
hedge-sd	0.2884	**21.7621**	**69.8531**	0.0067	**208.6723**	0.1796	**207.1898**	0.2895
intensity-mean	**3.9574**	**2.1687**	**1.3030**	**3.8936**	0.3634	**7.1007**	**1.7675**	**3.4010**
rawred-mean	**3.0212**	**2.5154**	**1.3864**	**3.5434**	0.2303	**8.3460**	1.5910	**2.8607**
rawblue-mean	**4.9499**	**2.5363**	**1.0718**	**4.1286**	0.8258	**5.7401**	**1.3864**	**4.0582**
rawgreen-mean	**3.4086**	**1.4650**	**1.4573**	**3.8519**	0.2640	**7.2421**	**2.4677**	**2.9034**
exred-mean	0.6573	0.3574	0.2632	**2.3650**	0.0949	**1.0620**	0.2845	0.5664
exblue-mean	**1.1291**	**1.1280**	0.3342	**3.9761**	0.1189	**1.7735**	0.4035	**1.4148**
exgreen-mean	**1.2551**	0.3515	0.7094	**1.7040**	0.1769	**2.4896**	0.3493	**1.5726**
value-mean	**4.7500**	**2.0473**	**1.0422**	**4.0145**	0.8030	**5.5291**	**1.3557**	**3.9385**
saturation-mean	0.4329	0.2982	**2.1785**	**3.7801**	0.6078	0.2884	0.0501	0.8251
hue-mean	**6.3530**	**1.3434**	**24.8247**	**28.2962**	**17.4598**	**14.6969**	0.4270	6.7460

"(1) region-centroidcol" attributes, had the highest (6.3530) and the lowest (0.1392) relevance weight in the definition of the clusters, respectively.

For the seven-cluster hard partition given by $MRDCA-RWL$ algorithm,

Table 15
Image data set: confusion matrix

	Classes						
Clusters	1-sky	2-cement	3-window	4-brick	5-grass	6-foliage	7-path
1	0	0	22	37	1	63	0
2	0	0	1	215	0	252	0
3	146	0	200	13	198	0	1
4	0	330	0	0	0	0	0
5	184	0	30	64	103	13	2
6	0	0	77	1	28	2	0
7	0	0	0	0	0	0	327

Table 14 shows (in bold) the dissimilarity matrices of most relevance weights in the definition of each cluster. For example, dissimilarity matrices computed taking into account only "(4) vegde-sd," "(6) hegde-sd," "(16) hue-mean," and "(3) vegde-mean" (in this order) are the most relevant in the definition of cluster 4 (cement).

4.2.4 Iris plant data set

This data set consists of three types (classes) of iris plants: iris setosa, iris versicolour and iris virginica. The three classes each have 50 instances (objects). One class is linearly separable from the other two; the latter two are not linearly separable from each other. Each object is described by four real-valued attributes: (1) sepal length, (2) sepal width, (3) petal length and (4) petal width.

The fuzzy clustering algorithms $NERF$ and $CARD-R$ were applied to the dissimilarity matrices obtained from this data set to obtain a three-cluster fuzzy partition. The three-cluster hard partitions obtained from the fuzzy partition were compared with the known a priori three-class partition. $NERF$ had 0.7294, 0.8922 and 10.67% for CR, $F-measure$, and $OERC$ indexes, respectively, whereas $CARD-R$ had 0.8856, 0.9599 and 4.00% for these indexes, respectively.

The hard clustering algorithms were applied to the dissimilarity matrices obtained from this data set to obtain a three-cluster hard partition. Table 16 shows the performance of the $SRDCA$, $MRDCA$, $MRDCA-RWL$ and $MRDCA-RWG$ algorithms on the iris data set according to CR, $F-measure$ and $OERC$ indexes, considering prototypes of cardinality $|G_k|$ =1, 2, 3, 5 and 10 ($k=1,2,3$).

For this data set, globally, the best performance was presented by $CARD-R$, $MRDCA-RWG$, $MRDCA-RWL$ and $SRDCA$, in this order. The worst performance was presented by $MRDCA$ and $NERF$, in this order. Note that

the performance was improved for $MRDCA$ and worsened for $SRDCA$ and $MRDCA-RWL$, with the increase of the cardinality of the prototypes. The performance of $MRDCA-RWL$ was not affected by the cardinality of the prototypes.

Table 16
Iris data set: CR, $F-measure$, and $OERC$ indexes

Indexes	$\|G_k\|$	$SRDCA$	$MRDCA$	$MRDCA-RWL$	$MRDCA-RWG$
CR	1	0.7455	0.6412	0.8680	0.8856
	2	0.7037	0.6412	0.8680	0.8856
	3	0.7302	0.6575	0.8507	0.8856
	5	0.7294	0.6575	0.8342	0.8856
	10	0.7436	0.6451	0.8681	0.8856
$F-measure$	1	0.8976	0.8465	0.9533	0.9599
	2	0.8782	0.8465	0.9533	0.9599
	3	0.8917	0.8600	0.9466	0.9599
	5	0.8922	0.8600	0.9398	0.9599
	10	0.8987	0.8535	0.9532	0.9599
$OERC$	1	10.00%	15.33%	4.67%	4.00%
	2	12.00%	15.33%	4.67%	4.00%
	3	10.67%	14.00%	5.33%	4.00%
	5	10.67%	14.00%	6.00%	4.00%
	10	10.00%	14.67%	4.67%	4.00%

Table 17 gives the vector of relevance weights, globally for all dissimilarity matrices (according to the best result given by $MRDCA-RWG$ algorithm with prototypes of cardinality 1) and locally for each cluster and dissimilarity matrix (according to the best result given by $MRDCA-RWL$ algorithm with prototypes of cardinality 1). Table 18 gives the confusion matrix of the three-cluster hard partition given by the $MRDCA-RWL$ algorithm with prototypes of cardinality 1.

Table 17
Iris data set: vectors of relevance weights

Data Matrix	$MRDCA-RWG$	$MRDCA-RWL$		
		Cluster 1	Cluster 2	Cluster 3
Sepal length	0.5523	0.4215	0.4423	0.4145
Sepal width	0.2971	0.5146	0.3555	0.0994
Petal length	**2.9820**	**2.3212**	**2.0378**	**7.3868**
Petal width	**2.0428**	**1.9861**	**3.1202**	**3.2822**

Table 18
Iris data set: confusion matrix

	Classes		
Clusters	1-Iris setosa	2-Iris versicolour	3-Iris virginica
1	50	0	0
2	0	3	46
3	0	47	4

Concerning the three-cluster partition given by $MRDCA - RWG$, dissimilarity matrices computed taking into account only "(3) petal length" or only "(4) petal width" attributes have the highest relevant weight. Thus the objects described by these dissimilarity matrices are closer to the prototypes of the clusters than are those described by dissimilarity matrices computed taking into account only "(1) sepal length" or "(2) sepal width" attributes.

For the three-cluster hard partition given by $MRDCA - RWL$ algorithm, Table 17 shows (in bold) the dissimilarity matrices of most relevance weights in the definition of each cluster. For example, dissimilarity matrices computed taking into account only "(3) Petal length" and "(4) Petal width" (in this order) are the most relevant in the definition of cluster 3 (Iris setosa), whereas dissimilarity matrices computed taking into account only "(4) Petal width" and "(3) Petal length" are the most relevant in the definition of cluster 2 (Iris versicolour).

4.2.5 Thyroid gland data set

This data set consists of three classes concerning the state of the thyroid gland: normal, hyperthyroidism and hypothyroidism. The classes (1, 2 and 3) have 150, 35 and 30 instances, respectively. Each object is described by five real-valued attributes: (1) T3-resin uptake test, (2) total serum thyroxin, (3) total serum triiodothyronine, (4) basal thyroid-stimulating hormone (TSH) and (5) maximal absolute difference in TSH value.

The fuzzy clustering algorithms $NERF$ and $CARD - R$ were applied to the dissimilarity matrices obtained from this data set to obtain a three-cluster fuzzy partition. The three-cluster hard partitions obtained from the fuzzy partition were compared with the known a priori three-class partition. $NERF$ had 0.4413, 0.7993 and 20.93% for CR, $F - measure$, and $OERC$ indexes, respectively, whereas $CARD - R$ had 0.2297, 0.7160 and 21.86% for these indexes, respectively.

The hard clustering algorithms were applied to the dissimilarity matrices obtained from this data set to obtain a three-cluster hard partition. Table 19 shows the performance of the $SRDCA$, $MRDCA$, $MRDCA - RWL$ and $MRDCA - RWG$ algorithms on the thyroid data set according to CR,

$F-measure$ and $OERC$ indexes, considering prototypes of cardinality $|G_k|$ =1, 2, 3, 5 and 10 ($k = 1, 2, 3$).

For this data set, globally, the best performance was presented by $MRDCA-RWL$, $MRDCA-RWG$, and $MRDCA$, in this order. The worst performance was presented by $CARD-R$, $NERF$ and $SRDCA$, in this order. In particular, $MRDCA-RWL$ with prototypes of cardinality 1 had the best and $SRDCA$ with prototypes of cardinality 10 had the worst performance, concerning these indexes. Note that the performance was stable for $MRDCA-RWG$ and was worsened for $MRDCA-RWL$, with the increase of the cardinality of the prototypes. Performance of $SRDCA$ was better with prototypes of cardinality 2, 3 and 5 and worst with prototypes of cardinality 10. Finally, the performance of $MRDCA$ was worst with prototypes of cardinality 2 and better with prototypes of cardinality 3, 5 and 10.

Table 19
Thyroid data set: CR, $F-measure$, and $OERC$ indexes

Indexes	$\|G_k\|$	$SRDCA$	$MRDCA$	$MRDCA-RWL$	$MRDCA-RWG$
CR	1	0.3577	0.5665	0.8776	0.5809
	2	0.5217	0.5484	0.8475	0.5484
	3	0.5217	0.5809	0.8328	0.5809
	5	0.6285	0.5974	0.8182	0.5809
	10	0.2014	0.5831	0.8185	0.5831
$F-measure$	1	0.7709	0.8551	0.9616	0.8614
	2	0.8408	0.8508	0.9525	0.8508
	3	0.8408	0.8614	0.9481	0.8614
	5	0.8820	0.8665	0.9437	0.8614
	10	0.6764	0.8602	0.9429	0.8602
$OERC$	1	24.65%	13.02%	3.72%	12.55%
	2	15.81%	13.48%	4.65%	13.48%
	3	15.81%	12.55%	5.11%	12.55%
	5	11.62%	12.09%	5.58%	12.55%
	10	35.34%	12.55%	5.58%	12.55%

Table 20 gives the vector of relevance weights globally for all dissimilarity matrices (according to the best result given by $MRDCA-RWG$ algorithm with prototypes of cardinality 1) and locally for each cluster and dissimilarity matrix (according to the best result given by $MRDCA-RWL$ algorithm with prototypes of cardinality 1). Table 21 gives the confusion matrix of the

three-cluster hard partition given by the $MRDCA - RWL$ algorithm with prototypes of cardinality 1.

Table 20
Thyroid data set: vectors of relevance weights

Data Matrix	$MRDCA - RWG$	$MRDCA - RWL$		
		Cluster 1	Cluster 2	Cluster 3
T3-resin uptake test	0.6546	0.2437	0.0599	**1.7284**
Total serum thyroxin	**1.3982**	0.4086	0.0933	**4.9804**
Total serum triiodothyronine	0.9716	0.8272	0.0488	**5.2643**
Basal thyroidstimulating hormone (TSH)	**1.1822**	**12.93**	**29.3203**	0.1350
Maximal absolute difference in TSH value	0.9509	0.9136	**124.6778**	0.1633

Table 21
Thyroid data set: confusion matrix

	Classes		
Clusters	1-Normal	2-Hyperthyroidism	3-Hypothyroidism
1	148	0	6
2	2	35	0
3	0	0	24

Concerning the three-cluster partition given by $MRDCA-RWG$, dissimilarity matrices computed taking into account only " (2) Total serum thyroxin" and "(1) T3-resin uptake test" attributes, had the highest (1.3982) and the lowest (0.6546) relevance weight in the definition of the clusters, respectively.

For the three-cluster hard partition given by $MRDCA - RWL$ algorithm, Table 20 shows (in bold) the dissimilarity matrices of most relevance weights in the definition of each cluster. For example, dissimilarity matrices computed taking into account only "(5) Maximal absolute difference in TSH value" and"(4) Basal thyroid-stimulating hormone (TSH)" (in this order) are the most relevant in the definition of cluster 2 (Hyperthyroidism), whereas dissimilarity matrices computed taking into account only "(3) Total serum triiodothyronine," "(2) Total serum thyroxin," and "(1) T3-resin uptake test" are the most relevant in the definition of cluster 3 (Hypothyroidism).

4.2.6 Wine data set

This data set consists of three types (classes) of wines grown in the same region in Italy, but derived from three different cultivars. The classes (1, 2, and 3) have 59, 71 and 48 instances, respectively. Each wine is described by 13 real-valued attributes representing the quantities of 13 components found in each of the three types of wines. These attributes are: (1) alcohol; (2) malic acid; (3) ash; (4) alkalinity of ash; (5) magnesium; (6) total phenols; (7) flavonoids; (8) non-flavonoid phenols; (9) proanthocyanins; (10) color intensity; (11) hue; (12) OD280/OD315 of diluted wines; and (13) proline.

The fuzzy clustering algorithms $NERF$ and $CARD-R$ were applied to the dissimilarity matrices obtained from this data set to obtain a three-cluster fuzzy partition. The three-cluster hard partitions obtained from the fuzzy partition were compared with the known a priori three-class partition. $NERF$ had 0.3539, 0.6986 and 31.46% for CR, $F-measure$, and $OERC$ indexes, respectively, whereas $CARD-R$ had 0.3808, 0.7227 and 26.97% for these indexes, respectively.

The hard clustering algorithms were applied to the dissimilarity matrices obtained from this data set to obtain a three-cluster hard partition. Table 22 shows the performance of the $SRDCA$, $MRDCA$, $MRDCA-RWL$ and $MRDCA-RWG$ algorithms on the wine data set according to CR, $F-measure$ and $OERC$ indexes, considering prototypes of cardinality $|G_k|$ =1, 2, 3, 5 and 10 ($k=1,2,3$).

For this data set, globally, the best performance was presented by $MRDCA$, $MRDCA-RWG$, $MRDCA-RWL$, and $CARD-R$, in this order. The worst performance was presented by $NERF$ and $SRDCA$, in this order. In particular, $MRDCA$ with prototypes of cardinality 5 or 10 had the best and $NERF$ had the worst performances, concerning these indexes. Note that the performance was worsened for $SRDCA$ and was improved for $MRDCA$, $MRDCA-RWL$, and $MRDCA-RWG$, with the increase of the cardinality of the prototypes.

Table 23 gives the vector of relevance weights globally for all dissimilarity matrices (according to the best result given by $MRDCA-RWG$ algorithm with prototypes of cardinality 10) and locally for each cluster and dissimilarity matrix (according to the best result given by $MRDCA-RWL$ algorithm with prototypes of cardinality 5). Table 24 gives the confusion matrix of the three-cluster hard partition given by the $MRDCA-RWL$ algorithm with prototypes of cardinality 5.

Concerning the three-cluster hard partition given by $MRDCA-RWG$, dissimilarity matrices computed taking into account only " (7) Flavonoids" and "(3) Ash" attributes, had the highest and the lowest relevance weight in the definition of the clusters, respectively.

For the three-cluster hard partition given by $MRDCA-RWL$ algorithm, Table 23 shows (in bold) the dissimilarity matrices of most relevance weights in the definition of each cluster. For example, dissimilarity matrices computed taking into account only "(7) Flavonoids," "(12) OD280/OD315 of diluted wines," "(13) Proline," "(11) Hue," and "(6) Total phenols" (in this order) are the most relevant in the definition of cluster 1 (Wine type 1).

In conclusion, for these UCI machine learning data sets, the best performance was presented by $MRDCA-RWL$, $MRDCA-RWG$, $MRDCA$,

Table 22
Wine data set: CR, $F-measure$, and $OERC$ indexes

Indexes	$\lvert G_k\rvert$	$SRDCA$	$MRDCA$	$MRDCA-RWL$	$MRDCA-RWG$
CR	1	0.3749	0.7263	0.7407	0.7548
	2	0.3711	0.8297	0.7702	0.8150
	3	0.3749	0.8319	0.7553	0.8319
	5	0.3711	0.8804	0.7712	0.8185
	10	0.3711	0.8804	0.7702	0.8348
$F-measure$	1	0.7204	0.9024	0.9077	0.9138
	2	0.7147	0.9435	0.9195	0.9372
	3	0.7204	0.9429	0.9136	0.9429
	5	0.7147	0.9603	0.9194	0.9371
	10	0.7147	0.9603	0.9195	0.9430
$OERC$	1	29.21%	9.55%	8.98%	8.42%
	2	29.77%	5.61%	7.86%	6.17%
	3	29.21%	5.61%	8.42%	5.61%
	5	29.77%	3.93%	7.86%	6.17%
	10	29.77%	3.93%	7.86%	5.61%

Table 23
Wine data set: vectors of relevance weights

Data Matrix	$MRDCA-RWG$	$MRDCA-RWL$		
		Cluster 1	Cluster 2	Cluster 3
Alcohol	**1.1425**	0.8661	0.6262	**1.4453**
Malic acid	0.7764	0.4632	**1.6446**	0.5761
Ash	0.5881	0.8849	0.4594	0.4902
Alkalinity of ash	0.6648	0.6879	0.5142	0.5693
Magnesium	0.5914	0.6026	0.4912	0.4559
Total phenols	**1.2453**	**1.0469**	**1.5799**	**1.0290**
Flavonoids	**2.5725**	**4.4077**	**2.5278**	**2.5297**
Non-flavonoid phenols	0.7232	0.5121	**1.6356**	0.6200
Proanthocyanins	0.7954	0.8521	0.8685	0.6673
Color intensity	0.9707	0.2827	**1.5606**	**4.6254**
Hue	0.9462	**1.3060**	**1.2557**	0.5543
OD280/OD315 of diluted wines	**1.7677**	**3.3920**	**1.4217**	0.9359
Proline	**1.6284**	**2.6922**	0.5292	**3.6505**

and $CARD-R$, in this order, according to CR, $F-measure$, and $OERC$ indexes. The worst performance was presented by $NERF$ and $SRDCA$, in this

Table 24
Wine data set: confusion matrix

	Classes		
Clusters	Wine type 1	Wine type 2	Wine type 3
1	59	8	0
2	0	57	0
3	0	6	48

order. Moreover, when the cardinality of the prototypes was increased, in the majority of the data sets, the performance was worsened for $MRDCA-RWL$, was worsened or stable for $MRDCA-RWG$ and $SRDCA$, and was improved for $MRDCA$.

4.3 Symbolic data sets

Symbolic data have been mainly studied in the area of Symbolic Data Analysis (SDA), where very often an object represents a group of individuals and the variables used to describe it need to assume a value which express the variability inherent to the description of a group. Thus, in SDA a variable can be interval-valued (it may assume as value an interval from a set of real numbers), set-valued (it may assume as value a set of categories), list-valued (it may assume as value an ordered list of categories) or even histogram-valued (it may assume as value an histogram). SDA aims to introduce new methods as well as to extend classical data analysis techniques (clustering, factorial techniques, decision trees, etc.) in order to manage these kinds of data (sets of categories, intervals, histograms), called symbolic data [**?**,**?**,**?**,**?**]. SDA is then an area related to multivariate analysis, pattern recognition and artificial intelligence.

This paper considers the following data sets described by symbolic (interval-valued and/or histogram-valued) variables: car and ecotoxicology data sets (http://www.info.fundp.ac.be/asso/) as well as horse data set (http://www.ceremade.dauphine.fr/~ touati/sodas-pagegarde.htm). Car and ecotoxicology datasets are described by a data matrix of "objects $\times$ interval-valued attributes". Horse dataset is described by a data matrix of "objects $\times$ attributes" where the attributes are interval-valued and histogram-valued.

Let $E = \{e_1, \ldots, e_n\}$ be a set of n objects described by p symbolic variables. Each object $e_i\,(i = 1, \ldots, n)$ is represented as a vector $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})$ of symbolic features values $x_{ij}\,(j = 1, \ldots, p)$. If the j-th symbolic variable is interval-valued, the symbolic feature value is an interval, i.e., $x_{ij} = [a_{ij}, b_{ij}]$ with $x_{ij} \in \mathbb{R}$. However, if the j-th symbolic variable is histogram-valued, the symbolic feature value is an histogram, i.e., $x_{ij} = (D_j, \mathbf{q}_{ij})\,(i = 1, \ldots, n; j = 1, \ldots, p)$ where D_j (the domain of the variable j) is a set of categories and

$\mathbf{q}_{ij} = (q_{ij1}, \ldots, q_{ijH_j})$ is a vector of weights.

A number of dissimilarity functions have been introduced in the literature for symbolic data in order to compare symbolic features values [**?**,**?**]. In this paper, we will consider suitable dissimilarities functions in order to compare a pair of objects $(e_i, e_l)\,(i, l = 1, \ldots, n)$ according to the pair $(x_{ij}, x_{lj})\,(j = 1, \ldots, p)$ of symbolic feature values given by the j-th symbolic variable.

If the j-th symbolic variable is interval-valued, the dissimilarity between the pair of intervals $x_{ij} = [a_{ij}, b_{ij}]$ and $x_{lj} = [a_{lj}, b_{lj}]$ will be computed according to the function given in [**?**]

$$d_j(x_{ij}, x_{lj}) = [max(b_{ij}, b_{lj}) - min(a_{ij}, a_{lj})] - \left[\frac{(b_{ij} - a_{ij}) + (b_{lj} - a_{lj})}{2}\right] \quad (37)$$

If the j-th symbolic variable is histogram-valued, the dissimilarity between the pair of histograms $x_{ij} = (D_j, \mathbf{q}_{ij}) = (D_j, (q_{ij1}, \ldots, q_{ijH_j}))$ and $x_{lj} = (D_j, \mathbf{q}_{lj}) = (D_j, (q_{lj1}, \ldots, q_{ljH_j}))$ will be computed according to the function given in [**?**]

$$d_j(x_{ij}, x_{lj}) = 1 - \sum_{m=1}^{H_j} \sqrt{\left(\frac{q_{ijm}}{\sum_{m=1}^{H_j} q_{ijm}}\right)\left(\frac{q_{ljm}}{\sum_{m=1}^{H_j} q_{ljm}}\right)} \quad (38)$$

Note that despite the usefulness of these dissimilarity functions to compare interval-valued or histogram-valued symbolic data, they cannot be used in object-based clustering because they are not differentiable with respect to the prototype parameters.

Several dissimilarity matrices are obtained from these data matrices. Concerning car and fish datasets, one of these dissimilarity matrices has the cells which are the dissimilarities between pairs of objects computed taking into account simultaneously all the interval-valued attributes, i.e, given two objects e_i and e_l described, respectively by $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})$ and $\mathbf{x}_l = (x_{l1}, \ldots, x_{lp})$, the dissimilarity between them taking into account simultaneously all the interval-valued attributes is computed as

$$d(\mathbf{x}_i, \mathbf{x}_l) = \sum_{j=1}^{p} d_j(x_{ij}, x_{lj}) \quad (39)$$

where $d_j(x_{ij}, x_{lj})$ is given by equation (37).

Horse data set is described by interval-valued as well as histogram-valued

attributes and so it is not suitable to produce a dissimilarity matrix having the cells which are the dissimilarities between pairs of objects computed taking into account simultaneously all the attributes.

For all symbolic datasets, all the others dissimilarity matrices have the cells which are the dissimilarities between pairs of objects computed taking into account only a single attribute.

For car and fish datasets, $NERF$ and $SRFCM$ were performed on the dissimilarity matrix which has the cells that are the dissimilarities between pairs of objects computed taking into account simultaneously all the attributes. For all datasets, $CARD-R$, $MRFCM$, $MRFCM-RWL$ and $MRFCM-RWG$ were performed simultaneously on all dissimilarity matrices which have the cells that are the dissimilarities between pairs of objects computed taking into account only a single attribute.

All dissimilarity matrices were normalized according to their overall dispersion [37] to have the same dynamic range. This means that each dissimilarity $d(e_k, e_{k'})$ in a given dissimilarity matrix has been normalized as $\frac{d(e_k,e_{k'})}{T}$ where $T = \sum_{k=1}^{n} d(e_k, g)$ is the overall dispersion and $g = e_l \in E = \{e_1, \ldots, e_n\}$is the overall prototype, which is computed according to $l = argmin_{1 \leq h \leq n} \sum_{k=1}^{n} d(e_k, e_h)$.

Each relational fuzzy clustering algorithm is run (until the convergence to a stationary value of the adequacy criterion) 100 times and the best result is selected according to the adequacy criterion. The parameters m, T and ϵ were set, respectively, equal to 2, 350 and 10^{-10}. The hard cluster partitions obtained from these fuzzy clustering methods were compared with the known a priori class partition. The comparison criteria used were the corrected Rand index (CR) and the overall error rate of classification ($OERC$). The CR and $OERC$ indexes were calculated for the best result.

4.3.1 Car data set

This dataset consists of four types (classes) of cars. The classes (1-Utility, 2-Sedan, 3-Sports and 4-Luxury) have, respectively, 10, 8, 8 and 7 instances. Each car is described by 8 interval-valued attributes : (1) price; (2) engine capacity; (3) top speed; (4) acceleration; (5) step; (6) length; (7) width and (8) height.

The fuzzy clustering algorithms were applied to the dissimilarity matrices obtained from this dataset in order to obtain a 4-cluster fuzzy partition. The 4-cluster hard partitions obtained from the fuzzy partition were compared with the known a priori 4-class partition. Table 34 shows the performance of the $SRFCM$, $MRFCM$, $MRFCM-RWL$ and $MRFCM-RWG$ algorithms on the car dataset according to CR and $OERC$ indexes, considering prototypes

of cardinality $|G_k|$ =1, 2, and 13 ($k = 1, 2, 3, 4$). $NERF$ had, respectively, 0.2543 and 51.52% for CR and $OERC$ indexes, whereas $CARD - R$ had, respectively, 0.5257 and 21.21% for these indexes.

For this dataset, globally, the best performance was presented by $MRFCM - RWG$, $MRFCM$, $MRFCM - RWL$ and $CARD - R$, in this order. The worst performance was presented by $NERF$ and $SRFCM$, in this order. In particular, $MRFCM$ (with prototypes of cardinality 2 or 3), $MRFCM - RWL$ (with prototypes of cardinality 2 or 3) and $MRFCM - RWG$ (whatever the cardinality of the prototypes) had the best and $SRFCM$ (with prototypes of cardinality 2 or 3) had the worst performance, concerning these indexes. Moreover, the performance of $MRFCM - RWG$, $MRFCM$ and $MRFCM - RWL$, according to these indices, were also superior to the performance presented by object-based fuzzy clustering algorithms with adaptive Euclidean distances which learns a relevance weight globally for each variable ($CR = 0.499$ and $OERC = 0.212$ [**?**]) or locally for each variable and each cluster ($CR = 0.526$ and $OERC = 0.212$ [**?**]).

Table 25
Car data set: CR and $OERC$ indexes

Indexes	$\|G_k\|$	$SRFCM$	$MRFCM$	$MRFCM - RWL$	$MRFCM - RWG$
CR	1	0.2584	0.5889	0.5791	0.6142
	2	0.2373	0.6142	0.6142	0.6142
	3	0.2373	0.6142	0.6142	0.6142
$OERC$	1	48.48%	18.18%	18.18%	15.15%
	2	45.45%	15.15%	15.15%	15.15%
	3	45.45%	15.15%	15.15%	15.15%

Table 35 gives the vector of relevance weights globally for all dissimilarity matrices (according to the result given by $MRFCM - RWG$ algorithm with prototypes of cardinality 2) and locally for each cluster and dissimilarity matrix (according to the result given by $MRFCM - RWL$ algorithm with prototypes of cardinality 2). Concerning the 4-cluster partition given by $MRFCM - RWG$, dissimilarity matrices computed taking into account only, respectively, "(3) Top speed" and "(8) Height" attributes, had, respectively, the highest and the lowest relevance weight in the definition of the fuzzy clusters.

Table 36 gives the confusion matrix of the 4-cluster hard partition obtained from the 4-cluster fuzzy partition given by the $MRFCM - RWL$ algorithm with prototypes of cardinality 2.

For the 4-cluster fuzzy partition given by $MRFCM - RWL$ algorithm, Table 35 shows (in bold) the dissimilarity matrices of most relevance weights in the

Table 26
Car data set: vectors of relevance weights

Data Matrix	$MRFCM-RWG$	$MRFCM-RWL$			
		Cluster 1	Cluster 2	Cluster 3	Cluster 4
Price	1.0669	0.7409	0.6685	**2.6030**	**1.3447**
Engine capacity	1.0072	0.8587	0.7948	**1.3894**	**1.1561**
Top speed	1.0934	**1.1680**	**1.4221**	**1.1149**	**1.0450**
Acceleration	0.9529	**1.5854**	**1.3830**	0.6675	0.8053
Step	0.9098	0.5988	0.7489	0.8958	0.8269
Length	1.0688	**1.6487**	**1.2328**	0.7401	0.8198
Width	1.0149	0.9794	**1.1231**	0.8211	0.8915
Height	0.9048	0.8775	0.9226	0.6822	**1.2643**

Table 27
Car data set: confusion matrix

Data Matrix	Classes			
	1-Utility	2-Sedan	3-Sports	4-Luxury
Cluster 1	0	0	0	7
Cluster 2	0	1	6	0
Cluster 3	8	0	0	0
Cluster 4	2	7	2	0

definition of each cluster. For example, dissimilarity matrices computed taking into account only "(3) Top speed", "(4) Acceleration", "(6) Length" and "(7) Width" (in this order) are the most relevant in the definition of cluster 2 (Sports), whereas dissimilarity matrices computed taking into account only "(1) Price", "(2) Engine capacity" and "(3) Top speed" are the most relevant in the definition of cluster 3 (Utility).

4.3.2 Ecotoxicology data set

This data set concerns 12 species (classes) of fresh water fish,with each species described by 13 interval-valued attributes. These species are grouped into four a priori classes of unequal sizes according to diet: two classes (1-Carnivorous and 2-Detritivorous) of size 4 and two classes (3-Omnivorous and 4-Herbivorous) of size 2. Each fish is described by 13 interval-valued attributes : (1) length; (2) weight; (3) muscle; (4) intestine; (5) stomach; (6) gills; (7) liver; (8) kidneys; (9) liver/muscle; (10) kidneys/muscle; (11) gills/muscle; (12) intestine/muscle and (13) stomach/muscle.

The fuzzy clustering algorithms were applied to the dissimilarity matrices obtained from this dataset in order to obtain a 4-cluster fuzzy partition. The 4-cluster hard partitions obtained from the fuzzy partition were compared with the known a priori 4-class partition. Table 28 shows the performance of the $SRFCM$, $MRFCM$, $MRFCM-RWL$ and $MRFCM-RWG$ algorithms on the ecotoxicology dataset according to CR and $OERC$ indexes,

considering prototypes of cardinality $|G_k|$ =1, and 2 ($k = 1, 2, 3, 4$). $NERF$ had, respectively, -0.1401 and 58.33% for CR and $OERC$ indexes, whereas $CARD - R$ had, respectively, 0.1606 and 33.33% for these indexes.

For this dataset, globally, the best performance was presented by $MRFCM$, $MRFCM - RWL$, $MRFCM - RWG$ and $CARD - R$, in this order. The worst performance was presented by $SRFCM$ and $NERF$, in this order. In particular, $MRFCM - RWG$, $MRFCM - RWL$ and $MRFCM - RWG$ (with prototypes of cardinality 3) had the best and $NERF$ and $SRFCM$ with prototypes of cardinality 1 had the worst performance, concerning these indexes. Moreover, the performance of $MRFCM$ and $MRFCM - RWL$ according to these indices, were also superior to the performance presented by object-based fuzzy clustering algorithms with adaptive Euclidean distances which learns a relevance weight globally for each variable ($CR = 0.201$ and $OERC = 0.416$ [**?**]) or locally for each variable and each cluster ($CR = 0.274$ and $OERC = 0.416$ [**?**]).

Table 28
Ecotoxicology data set: CR and $OERC$ indexes

Indexes	$\|G_k\|$	$SRFCM$	$MRFCM$	$MRFCM - RWL$	$MRFCM - RWG$
CR	1	-0.1401	0.2489	0.2245	0.2012
	2	0.0331	0.4880	0.4880	0.4880
$OERC$	1	58.33%	33.33%	41.67%	33.33%
	2	50.00%	16.67%	16.67%	16.67%

Table 29 gives the vector of relevance weights globally for all dissimilarity matrices (according to the best result given by $MRFCM - RWG$ algorithm with prototypes of cardinality 2) and locally for each cluster and dissimilarity matrix (according to the best result given by $MRFCM - RWL$ algorithm with prototypes of cardinality 2). Concerning the 4-cluster partition given by $MRFCM - RWG$, dissimilarity matrices computed taking into account only, respectively, "(6) Gills" and "(2) Weight" attributes, had, respectively, the highest and the lowest relevance weight in the definition of the fuzzy clusters.

Table 30 gives the confusion matrix of the 4-cluster hard partition obtained from the 4-cluster fuzzy partition given by the $MRFCM - RWL$ algorithm with prototypes of cardinality 2.

For the 4-cluster fuzzy partition given by $MRFCM - RWL$ algorithm, Table 29 shows (in bold) the dissimilarity matrices of most relevance weights in the definition of each cluster. For example, dissimilarity matrices computed taking into account only "(3) Miscle", "(8) Kidneys", "(5) Stomach", "(4) Intestine", "(2) Weight", "(7) Liver", "(6) Gills" and "(1) Lenght" (in this order) are the most relevant in the definition of cluster 1 (Omnivorous), whereas

Table 29
Ecotoxicology data set: vectors of relevance weights

Data Matrix	$MRFCM-RWG$	$MRFCM-RWL$			
		Cluster 1	Cluster 2	Cluster 3	Cluster 4
Length	0.9199	**1.1505**	0.4777	0.8768	**1.0519**
Weight	0.8405	**2.2442**	0.2848	0.8297	0.9325
Muscle	1.0994	**4.0608**	0.9308	0.9655	0.7518
Intestine	1.1005	**2.4509**	0.8445	0.8597	0.9294
Stomach	0.9743	**3.0993**	0.4379	0.7308	0.8355
Gills	1.2068	**1.7952**	0.7454	**1.1022**	**1.0023**
Liver	1.0512	**1.8715**	0.7555	0.7577	0.8216
Kidneys	0.8557	**3.3711**	0.4556	0.5570	0.9869
Liver/muscle	1.0762	0.4340	**2.5417**	0.8674	0.8886
Kidneys/muscle	0.9287	0.2902	**1.5922**	0.8046	**1.2491**
Gills/muscle	1.0028	0.1192	**3.8796**	**2.0996**	**2.2385**
Intestine/muscle	1.0688	0.2703	**2.0682**	**1.3147**	0.6815
Stomach/muscle	0.9430	0.2728	**2.5608**	**2.5265**	**1.2682**

Table 30
Ecotoxicology data set: confusion matrix

Data Matrix	Classes			
	1-Carnivorous	2-Detritivorous)	3-Omnivorous	4-Herbivorous
Cluster 1	0	0	2	0
Cluster 2	3	0	0	0
Cluster 3	1	3	0	0
Cluster 4	0	1	0	2

dissimilarity matrices computed taking into account only "(11) Gills/muscle", "(13) Stomach/muscle", "(9) Liver/muscle", "(12) Intestine/muscle and "(10) Kidneys/muscle" are the most relevant in the definition of cluster 2 (Carnivorous).

4.3.3 Horse dataset

This dataset describes 12 horses. Each horse is described by 7 interval-valued variables (Height at the withers (min), Height at the withers (max), Weight (min), Weight (max), Mares, Stallions, Birth) and 3 histogram-valued variables (Country, Robe and Aptitude). The horses are grouped into 4 a priori classes (1-(Racehorse, 2-Leisure Horse, 3-Poney and 4-Draft horse) which have, respectively, 4, 3, 3 and 2 instances.

The fuzzy clustering algorithms were applied to the dissimilarity matrices obtained from this dataset in order to obtain a 4-cluster fuzzy partition. The 4-cluster hard partitions obtained from the fuzzy partition were compared with the known a priori 4-class partition. Table 31 shows the performance of the $MRFCM$, $MRFCM-RWL$ and $MRFCM-RWG$ algorithms on the

horse dataset according to CR and $OERC$ indexes, considering prototypes of cardinality $|G_k|$ =1, and 2 ($k = 1, 2, 3, 4$). $CARD-R$ had, respectively, 0.2275 and 41.67% for these indexes.

For this dataset, globally, the best performance was presented by $MRFCM-RWL$, $MRFCM-RWG$ and $CARD-R$, in this order. The worst performance was presented by $MRFCM$. In particular, $MRFCM-RWL$ and $MRFCM-RWG$ (with prototypes of cardinality 1) had the best and $MRFCM$ with prototypes of cardinality 2 had the worst performance, concerning these indexes. Moreover, the performance of $MRFCM-RWL$ and $MRFCM-RWG$ (with prototypes of cardinality 1) according to these indices, were also superior to the performance presented by object-based hard clustering algorithms with adaptive Euclidean distances which learns a relevance weight globally for each variable ($CR = 0.209$ and $OERC = 0.417$ [**?**]) or locally for each variable and each cluster ($CR = 0.138$ and $OERC = 0.417$ [**?**]).

Table 31
Horse data set: CR and $OERC$ indexes

Indexes	$\|G_k\|$	$MRFCM$	$MRFCM-RWL$	$MRFCM-RWG$
CR	1	0.0946	0.3662	0.3662
	2	0.0041	0.2087	0.1851
$OERC$	1	41.67%	33.33%	33.33%
	2	50.00%	33.33%	41.67%

Table 32 gives the vector of relevance weights globally for all dissimilarity matrices (according to the best result given by $MRFCM-RWG$ algorithm with prototypes of cardinality 1) and locally for each cluster and dissimilarity matrix (according to the best result given by $MRFCM-RWL$ algorithm with prototypes of cardinality 1). Concerning the 4-cluster partition given by $MRFCM-RWG$, dissimilarity matrices computed taking into account only, respectively, "(6) Weight (min)" and "(2) Robe" attributes, had, respectively, the highest and the lowest relevance weight in the definition of the fuzzy clusters.

Table 33 gives the confusion matrix of the 4-cluster hard partition obtained from the 4-cluster fuzzy partition given by the $MRFCM-RWL$ algorithm with prototypes of cardinality 1.

For the 4-cluster fuzzy partition given by $MRFCM-RWL$ algorithm, Table 32 shows (in bold) the dissimilarity matrices of most relevance weights in the definition of each cluster. For example, dissimilarity matrices computed taking into account only "(10) Birth", "(6) Weight (min)", "(1) Country", "(9) Stallions", "(8) Mares" and "(7) Weight" (in this order) are the most relevant in the definition of cluster 3 (Poney).

Table 32
Horse data set: vectors of relevance weights

Data Matrix	$MRFCM-RWG$	$MRFCM-RWL$			
		Cluster 1	Cluster 2	Cluster 3	Cluster 4
Country	1.0180	**1.0780**	0.6030	**1.0794**	**1.2380**
Robe	0.8002	0.8390	0.5569	0.8133	0.8728
Ability	0.8082	**1.4118**	0.4054	0.5767	**1.2115**
Size (min)	0.9989	0.9849	**1.5925**	0.9109	0.8896
Size (max)	0.9453	0.9266	**1.2567**	0.9571	0.8914
Weight (min)	1.1582	0.9981	**1.7515**	**1.3624**	0.8971
Weight	1.0650	0.9671	**1.0576**	**1.0399**	**1.0534**
Mares	1.0745	0.9697	**1.0699**	**1.0525**	**1.0710**
Stallions	1.0801	0.9951	**1.0657**	**1.0663**	**1.0647**
Birth	1.1231	0.9208	**1.7366**	**1.4242**	0.8934

Table 33
Horse data set: confusion matrix

Data Matrix	Classes			
	1-(Racehorse	2-Leisure Horse	3-Poney	4-Draft horse
Cluster 1	2	0	0	2
Cluster 2	2	2	0	0
Cluster 3	0	0	3	0
Cluster 4	0	1	0	0

4.4 Time trajectories data sets

The authors consider phoneme and satellite time trajectories data sets. These data sets are available at http://www.math.univ-toulouse.fr/staph/npfda/npfda-datasets.html. To compare time trajectories, a "cross sectional-longitudinal" dissimilarity function proposed by D'Urso and Vichi was considered [35] [36]. The authors propose a compromise dissimilarity that is a combination of a cross-sectional dissimilarity, which compares the instantaneous position (trend) of each pair of trajectories, and two longitudinal dissimilarities, based on the concepts of velocity and acceleration of a time trajectory.

Let $\mathbf{x}_i = (x_i(t_1), \ldots, x_i(t_p))\,(i = 1, \ldots, n)$ the i-th time trajectory. The velocity of the i-th time trajectory is defined as $\mathbf{v}_i = (v_i(t_2), \ldots, v_i(t_p))\,(i = 1, \ldots, n)$, where $v_i(t_j) = \frac{x_i(t_j) - x_i(t_{j-1})}{t_j - t_{j-1}}$ $(j = 2 \ldots, p)$ is the velocity in the interval $[t_{j-1}, t_j)$ which measures the variation of the i-th time trajectory in $[t_{j-1}, t_j)$. The acceleration of the i-th time trajectory is defined as $\mathbf{a}_i = (a_i(t_3), \ldots, a_i(t_p))\,(i = 1, \ldots, n)$, where $a_i(t_j) = \frac{v_i(t_j) - v_i(t_{j-1})}{t_j - t_{j-2}}$ $(j = 3 \ldots, p)$ is the acceleration in the interval $[t_{j-2}, t_j)$.

The compromise dissimilarity between the i-th and the l-th time trajectories is defined as

$$d^2(i,l) = \alpha_1 \|\mathbf{x}_i - \mathbf{x}_l\|^2 + \alpha_2 \|\mathbf{v}_i - \mathbf{v}_l\|^2 + \alpha_3 \|\mathbf{a}_i - \mathbf{a}_l\|^2 \qquad (40)$$

where $\|\mathbf{x}_i - \mathbf{x}_l\| = \sum_{j=1}^{p}(x_i(t_j) - x_l(t_j))^2$, $\|\mathbf{v}_i - \mathbf{v}_l\| = \sum_{j=2}^{p}(v_i(t_j) - v_l(t_j))^2$ and $\|\mathbf{a}_i - \mathbf{a}_l\| = \sum_{j=3}^{p}(a_i(t_j) - a_l(t_j))^2$.

In [35], the weights α of each dissimilarity-component are determined by considering a global objective criterion based on the maximization of the variance of the compromise dissimilarity. In this paper, they will be determined according to the relational clustering algorithm presented in sections **??** and **??**.

Note that because they are based on a single dissimilarity matrix, neither $NERF$ nor $SRDCA$ can be used to cluster time trajectories data sets compared to the "cross sectional-longitudinal" dissimilarity function proposed by D'Urso and Vichi [35] [36].

4.5 Phoneme data set

This data set is a part of the original one that can be found at http://www-stat.stanford.edu/ tibs/ElemStatLearn/. It consists of five phonemes (classes): "sh," "iy," "dcl," "aa," and "ao". The five classes each have 400 instances (objects). Each object (time trajectory) is described as $(\mathbf{x}_i, y_i)$ $(i = 1 \ldots, n)$, where y_i gives the class membership (phonemes) whereas $\mathbf{x}_i = (x_i(t_1), \ldots, x_i(t_{150}))$ is the i-th discretized functional data corresponding to the discretized log-periodograms.

From the original phoneme data set the authors had obtained initially two additional data sets corresponding to the velocity and acceleration of the discretized log-periodograms. Then, three relational data tables are obtained from these three satellite data sets (position, velocity and acceleration of the discretized log-periodograms) through the application of the squared Euclidean distance. All dissimilarity matrices were normalized according to their overall dispersion [37] to have the same dynamic range.

The fuzzy clustering algorithm $CARD - R$ was performed simultaneously on these 3 relational data tables (position , velocity, and acceleration of the discretized log-periodograms) to obtain a five-cluster fuzzy partition. The five-cluster hard partitions obtained from the fuzzy partition were compared with the known a priori five-class partition. $CARD - R$ had 0.1922, 0.4853 and 60.40% for the CR, $F - measure$, and $OERC$ indexes, respectively.

The hard clustering algorithms $MRDCA$, $MRDCA - RWL$ and $MRDCA - RWG$ were applied simultaneously on these 3 relational data tables to obtain a five-cluster hard partition. Table 34 shows the performance of the $MRDCA$,

$MRDCA-RWL$ and $MRDCA-RWG$ algorithms on the phoneme data set according to the CR, $F-measure$ and $OERC$ indexes, considering prototypes of cardinality $|G_k|$ =1, 2, 3, 5 and 10 ($k = 1, \ldots, 5$).

For this data set, globally, the best performance was presented by $MRDCA-RWL$, $MRDCA-RWG$ and $MRDCA$, in this order. The worst performance was presented by $CARD-R$. In particular, $MRDCA-RWG$ with prototypes of cardinality 10 had the best performance, concerning these indexes. Note that the performance was improved for $MRDCA$, $MRDCA-RWL$ and $MRDCA-RWG$, with the increase of the cardinality of the prototypes.

Table 34
Phoneme data set: CR, $F-measure$, and $OERC$ indexes

Indexes	$\|G_k\|$	$MRDCA$	$MRDCA-RWL$	$MRDCA-RWG$
CR	1	0.4366	0.5418	0.5216
	2	0.4284	0.5835	0.5964
	3	0.5317	0.6972	0.6937
	5	0.4812	0.7270	0.7225
	10	0.4698	0.7264	0.7277
$F-measure$	1	0.6496	0.7435	0.7441
	2	0.6714	0.7675	0.7708
	3	0.7331	0.8448	0.8433
	5	0.6501	0.8550	0.8525
	10	0.6484	0.8495	0.8492
$OERC$	1	38.65%	27.10 %	28.55%
	2	34.50%	26.30%	25.45%
	3	30.15 %	15.70%	16.00%
	5	36.25%	14.70%	14.95%
	10	39.15%	15.10%	15.15%

Table 35 gives the vector of relevance weights globally for all dissimilarity matrices (according to the best result given by $MRDCA-RWG$ algorithm with prototypes of cardinality 10) and locally for each cluster and dissimilarity matrix (according to the best result given by $MRDCA-RWL$ algorithm with prototypes of cardinality 5). Table 36 gives the confusion matrix of the five-cluster hard partition given by the $MRDCA-RWL$ algorithm with prototypes of cardinality 5.

Concerning the five-cluster hard partition given by $MRDCA-RWG$, dissimilarity matrices computed taking into account only " (1) Position" attribute

Table 35
Phoneme data set: vectors of relevance weights

Data Matrix	$MRDCA-RWG$	$MRDCA-RWL$				
		Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
Position	**2.1888**	**2.5936**	**1.5062**	**2.3235**	**2.1424**	**2.0930**
Velocity	0.6900	0.6458	0.8203	0.6451	0.7102	0.7091
Acceleration	0.6621	0.5969	0.8093	0.6670	0.6571	0.6736

Table 36
Phoneme data set: confusion matrix

	Classes				
Clusters	1-sh	2-iy	3-dcl	4-aa	5-ao
1	0	1	387	0	1
2	396	9	1	0	0
3	0	0	0	271	115
4	0	21	9	129	283
5	4	369	3	0	1

had the highest relevance weight in the definition of the clusters. Thus, the objects described by this dissimilarity matrix are closer to the prototypes of the clusters than are those described by velocity or acceleration dissimilarity matrix.

For the five-cluster hard partition given by $MRDCA-RWL$ algorithm, Table 23 shows (in bold) the dissimilarity matrices of the most relevance weights in the definition of each cluster. For all clusters, position dissimilarity matrix has the highest relevant weight, thus the objects described by this dissimilarity matrix are closer to the respective prototypes of these clusters than are those described by velocity or acceleration dissimilarity matrices.

4.6 Satellite data set

This data set concerns $n = 472$ radar waveforms. The data were registered by the Topex/Poseidon satellite upon the Amazon River. Each object (time trajectory) is represented by its discretized wave version $\mathbf{x}_i = (x_i(t_1), \ldots, x_i(t_{70}))$ $(i = 1, \ldots, 472)$. Each wave is linked with the kind of ground treated by the satellite, and the aim is to use these waveforms for altimetric and hydrological purpose on the Amazonian basin.

From the original satellite data set, the authors obtained initially 2 additional data sets corresponding to the velocity and acceleration of the radar waveforms. Then, 3 relational data tables are obtained from these 3 satellite data sets (position, velocity and acceleration of the radar waveforms) through the application of the squared Euclidean distance. All dissimilarity matrices were normalized according to their overall dispersion [37] to have the same dynamic

range.

The clustering algorithm has been performed simultaneously on these 3 relational data tables (position, velocity and acceleration of the radar waveforms) to obtain a partition in $K = \{1, \ldots, 10\}$. For a fixed number of clusters K, the clustering algorithm is run 100 times and the best result according to the adequacy criterion is selected.

To determine the number of cluster, the authors used the approach described by [38], which consists of the choice of the peaks on the graph of the "second-oder differences" of the clustering criterion (equation (**??**)): $J^{(K-1)} + J^{(K+1)} - 2J^{(K)}$, $K = 2, \ldots, 9$. According to this approach, the number of clusters was fixed as 7. Algorithm $MRDCA - RWG$ gives 7 clusters with cardinality of 49, 55, 45, 79, 32, 149 and 63, while algorithm $MRDCA - RWL$ gives 7 clusters with cardinality of 38, 84, 61, 97, 62, 92 and 38. For both algorithms, the prototypes have cardinality 5.

Table 37 gives the vector of relevance weights globally for each dissimilarity matrix (according to the algorithm $MRDCA - RWG$) and locally for each cluster and dissimilarity matrix (according to the algorithm $MRDCA - RWL$). Concerning the seven-cluster partition given by $MRDCA - RWG$, position dissimilarity matrix has the highest relevant weight.

Table 37
Satellite data set: vectors of relevance weights

Data Matrix	$MRDCA - RWG$	$MRDCA - RWL$						
		Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	Cluster 6	Cluster 7
Position	**1.4309**	**3.4372**	0.7862	**1.6756**	**2.3028**	**2.6052**	**2.1688**	0.7543
Velocity	0.8447	0.5974	**1.1387**	0.7634	0.6757	0.6460	0.6997	**1.0608**
Acceleration	0.8272	0.4869	**1.1168**	0.7817	0.6425	0.5940	0.6588	**1.2496**

For clusters 1, 3, 4, 5 and 6 of the seven-cluster partition given by $MRDCA - RWL$, position dissimilarity matrix had the highest relevant weight, while for cluster 2, velocity and acceleration dissimilarity matrices, in this order, had the highest relevant weights. Finally, for cluster 7, acceleration and velocity dissimilarity matrices, in this order, had the highest relevant weights.

Figure 1 shows selected curves from the original satellite data set (position) belonging to each of the seven clusters. The 5 curves in the prototype of each cluster (1 and 7) are drawn in bold. This figure shows clearly the difference between the clusters.

Figures 2-4 show selected curves from the original satellite data set (position) as well as from the additional data sets (velocity and acceleration) belonging to clusters 1 (where position dissimilarity matrix had the highest relevant weight among the seven clusters) and 7 (where acceleration and velocity dissimilarity matrices were more relevant than position dissimilarity matrix). These figures

illustrate clearly why position is the most relevant dissimilarity matrix for cluster 1 whereas acceleration and velocity are the most relevant dissimilarity matrix for cluster 7. In these figures, the 5 curves in the prototype of each cluster (1 and 7) were also drawn in bold.

5 Concluding remarks

This paper extended the dynamic clustering algorithm for relational data ($SRDCA$) into hard clustering algorithms ($MRDCA-RWL$ and $MRDCA-RWG$) that are able to partition objects taking into account simultaneously their relational descriptions given by multiple dissimilarity matrices. These matrices have been generated using different sets of variables and dissimilarity functions. These algorithms are designed to furnish a partition and a prototype for each cluster as well as a relevance weight for each dissimilarity matrix by optimizing an adequacy criterion that measures the fitting between clusters and their representatives. As a particularity of these clustering algorithms, they assume that the prototype of each cluster is a subset (of fixed cardinality) of the set of objects.

For each algorithm, the paper gives the solution for the best prototype of each cluster, the best relevance weight of each dissimilarity matrix as well as the best partition, according to the clustering criterion. Moreover, the time complexity and the convergence properties of $MRDCA-RWL$ and $MRDCA-RWG$ are also presented. Concerning the relevance weights, they change at each algorithm iteration and can either be the same for all clusters or different from one cluster to another. Moreover, they are determined automatically in such a way that the closer to the prototype the objects of a given dissimilarity matrix of a given cluster are, the higher is the relevance weight of this dissimilarity matrix on this cluster.

The usefulness of these partitioning relational hard clustering algorithms was shown with data sets (synthetic and from UCI machine learning repository) described by real-valued variables as well as with time trajectory data sets. The accuracy of the results furnished by $MRDCA-RWL$ and $MRDCA-RWG$ algorithms on these data sets was assessed by the corrected Rand index, the F-measure and the overall error rate of classification.

Concerning the synthetic data sets, the performance of $MRDCA-RWL$ and $MRDCA-RWG$ depends on the dispersion of the variables that describes the objects. In comparison with the algorithms $NERF$ and $SRDCA$, which perform on a single dissimilarity matrix, $MRDCA-RWL$ was clearly superior in the synthetic data sets where the variance was different between the variables whereas $MRDCA-RWG$ was clearly superior only in the synthetic

data sets where the variance was different between the variables but almost the same from one class to another.

Moreover, for the UCI machine learning data sets, the best performance was presented by $MRDCA-RWL$, $MRDCA-RWG$, $MRDCA$ and $CARD-R$, in this order. The worst performance was presented by $NERF$ and $SRDCA$ (algorithms that perfoms on a single dissimilarity matrix). Moreover, when the cardinality of the prototypes was increased, in the majority of these data sets, the performance was worsened for $MRDCA - RWL$, was worsened or stable for $MRDCA - RWG$ and $SRDCA$ and was improved for $MRDCA$.

Phoneme and satellite time trajectory data sets compared through a "cross sectional-longitudinal" dissimilarity function also have been considered. Because this dissimilarity function, when applied to a data set, produces three dissimilarity matrices corresponding to the comparison of the trajectories according to the their trend, velocity and acceleration, only relational clustering algorithms that are able to manage multiple dissimilarity matrices can be considered. Thus, for the phoneme time trajectory data set, the best performance was presented by $MRDCA - RWL$, $MRDCA - RWG$ and $MRDCA$, in this order. The worst performance was presented by $CARD - R$. Moreover, the performance of $MRDCA$, $MRDCA - RWL$ and $MRDCA - RWG$ was improved with the increase of the cardinality of the prototypes. Finally, the usefulness of the algorithms $MRDCA - RWL$ and $MRDCA - RWG$ have also been illustrated with the study of the satellite time trajectory data set.

References

[1] A. K. Jain, M.N. Murty, P.J. Flynn, Data Clustering: A Review, ACM Computing Surveys 31 (3) (1999) 264–323

[2] R. Xu, D. Wunsch, Survey of Clustering Algorithms, IEEE Transactions on Neural Networks 16 (3) (2005) 645–678

[3] P.H. Sneath, R.R. Sokal, Numerical Taxonomy. Freeman, San Francisco, 1973

[4] T. Zhang, R. Ramakrishnan, and M. Livny, BIRCH: An efficient data clustering method for very large databases, in Proc. ACM SIGMOD Conf. Management of Data, 1996, pp. 103114.

[5] S. Guha, R. Rastogi, and K. Shim, CURE: An efficient clustering algorithm for large databases, in Proc. ACM SIGMOD Int. Conf. Management of Data, 1998, pp. 7384.

[6] G. Karypis, E. Han, and V. Kumar, Chameleon: Hierarchical clustering using dynamic modeling, IEEE Computer 32 (8) (1999) 68–75

[7] S. Guha, R. Rastogi, and K. Shim, ROCK: A robust clustering algorithm for categorical attributes, Information Systems 25 (5) (2000) 345–366

[8] G.N. Lance, W.T. Williams, Note on a new information statistic classification program, The Computer Journal 11 (1968) 195–197

[9] K.C. Gowda, G. Krishna, Disaggregative clustering using the concept of mutual nearest neighborhood, IEEE Transactions on Systems, Man, and Cybernetics 8 (1978) 888–895

[10] L. Kaufman, P.J. Rousseeuw, Finding Groups in Data, Wiley, New York, 1990

[11] A. Guenoche, P. Hansen, B. Jaumard, Efficient algorithms for divisive hierarchical clustering, Journal of Classification 8 (1991) 5–30.

[12] M. Chavent, A monotetic clustering method, Pattern Recognition Letters 19 (1998) 989–996

[13] E. Forgy, Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications, Biometrics 21 (1965) 768–780

[14] Z. Huang, Extensions to the K-means algorithm for clustering large data sets with categorical values, Data Mining and Knowledge Discovery 2 (1998) 283–304

[15] T. Kanungo, D. Mount, N. Netanyahu, C. Piatko, R. Silverman, and A. Wu, An efficient K-means clustering algorithm: Analysis and implementation, IEEE Transactions in Pattern Analysis Machine Intelligence 24 (7) (2000) 881–892

[16] P. Hansen and N. Mladenoviae, J-means: A new local search heuristic for minimum sum of squares clustering, Pattern Recognition 34 (2001) 405–413

[17] M. Su and C. Chou, A modified version of the K-means algorithm with a distance based on cluster symmetry, IEEE Transactions on Pattern Analysis and Machine Intelligence 23 (6) (2001) 674–680

[18] J. C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York, 1981

[19] F. Hoeppner, F. Klawonn, and R. Kruse, Fuzzy Cluster Analysis: Methods for Classification, Data Analysis, and Image Recognition. Wiley, New York, 1999.

[20] R. Hathaway, J. Bezdek, and Y. Hu, Generalized fuzzy c-means clustering strategies using L_p norm distances, IEEE Transactions on Fuzzy Systems 8 (5) (2000) 576–582

[21] M. Hung and D. Yang, An efficient fuzzy c-means clustering algorithm, in Proc. IEEE Int. Conf. Data Mining, 2001, 225–232.

[22] J. Kolen and T. Hutcheson, Reducing the time complexity of the fuzzy c-means algorithm, IEEE Transactions on Fuzzy Systems 10 (2) (2002) 263–267

[23] Y. Lechevallier, Optimisation de quelques criteres en classification automatique et application a l'etude des modifications des proteines seriques en pathologie clinique. Thèse de 3eme cycle. Universite Paris-VI, 1974

[24] F.A.T. De Carvalho, M.Csernel, Y. Lechevallier, Pattern Recognition Letters 30 (2009) 10371045

[25] J.W. Davenport, R.J. Hathaway, J.C. Bezdek, Relational duals of the c-means algorithms, Pattern Recognition 22 (1989) 205–212

[26] R.J. Hathaway, J.C. Bezdek, Nerf c-means: non-Euclidean relational fuzzy clustering, Pattern Recognition 27 (3) (1994) 429437

[27] H. Frigui, C. Hwanga, F. C.-H. Rhee, Clustering and aggregation of relational data with applications to image database categorization, Pattern Recognition, 40 (11) (2007) 3053 3068

[28] W. Pedrycz, Collaborative fuzzy clustering, Pattern Recognition Letters, 23, (2002) 675–686

[29] E. Diday, G. Govaert, Classification Automatique avec Distances Adaptatives, R.A.I.R.O. Informatique Computer Science 11 (4) (1977) 329–349.

[30] E. Diday, J.C. Simon, Clustering analysis, in K.S. Fu (ed), Digital Pattern Classification, Springer, Berlin, 1976, 47–94.

[31] L. Hubert, P. Arabie, Comparing partitions, Journal of Classification 2 (1985) 193–218

[32] C.J. van Rijisbergen, Information retrieval, Butterworth-Heinemann, London, 1979.

[33] L. Breiman, J. Friedman, C.J. Stone,R.A. Olshen, Classification and Regression Trees, Chapman and Hall/CRC, Boca Raton, 1984

[34] G. W. Milligan, Clustering Validation: results and implications for applied analysis, in P. Arabie, L. Hubert, G. De Soete (eds), Clustering and Classification, Word Scientific, Singapore, 341–375, 1996

[35] P. D'Urso and M. Vichi, Dissimilarities between trajectories of a three-way longitudinal data set, in A. Rizzi, M. Vichi, H.-H. Bock, Advances in Data Science and Classification, Springer, Berlin, 585–592, 1998

[36] P. D'Urso, Dissimilarity measures for time trajectories, Journal of Italian Statistical Society 1 (3) (2000) 53–83

[37] M. Chavent, Normalized k-means clustering of hyper-rectangles, in: Proceedings of the XI International Symposium of Applied Stochastic Models and Data Analysis (ASMDA 2005), Brest, France, 2005, pp. 670–677

[38] A. Da Silva, Analyse de données évolutives: application aux données d'usage Web, Thèse de Doctorat, Université Paris-IX Dauphine, 2009.