2009 Ninth International Conference on Intelligent Systems Design and Applications

Fast Evolutionary Algorithms for Relational Clustering

Danilo Horta and Ricardo J. G. B. Campello
Department of Computer Sciences
University of São Paulo - São Carlos, SP, Brazil
{horta,campello}@icmc.usp.br

***Abstract*—This paper is concerned with the computational efficiency of clustering algorithms when the data set to be clustered is described by a proximity matrix only (relational data) and the number of clusters must be automatically estimated from such data. Two relational versions of an evolutionary algorithm for clustering are derived and compared against two systematic (repetitive) approaches that can also be used to automatically estimate the number of clusters in relational data. Exhaustive experiments involving six artificial and two real data sets are reported and analyzed.**

I. Introduction

There exist at least two elementary ways in which a data set is defined for clustering tasks. The first one takes place when the data set $O = \{o_1, o_2, \ldots, o_n\}$, composed of n data objects, is described by means of a set of attribute vectors $X = \{x_1, x_2, \ldots, x_n\}$, where x_i is the attribute vector of the ith data object, o_i. Alternatively, the data set O can be described only by similarity or dissimilarity relations between its objects; for instance, through a relational matrix $R = [r_{ij}]_{n \times n}$, in which r_{ij} is the value of similarity or dissimilarity between objects o_i and o_j [1]. In this case, the data are referred to as relational data and the clustering approaches capable of processing them are so-called relational clustering algorithms. It turns out that these relational algorithms are of great importance to private data since they do not need access to the object attributes. For instance, a data set containing information about a bank's clients can be shared through a relational matrix, which hides all the information except the (dis)similarity between the clients. Moreover, there are several fields of knowledge, such as numerical taxonomy, seismic engineering, expert systems design, document retrieval, management, industrial engineering, and social sciences, in which relational data is commonly and naturally encountered [2], [3]. For instance, a relational matrix whose values represent the subjective dissimilarities between 11 sciences is reported in [4], which also describes an example involving the dissimilarities between 12 countries assigned subjectively by a group of political science students. Another relevant aspect of relational algorithms stems from applications involving mixed data types (e.g., categorical and numerical). By means of suitable (dis)similarity functions, a data set with mixed data types can be described by a relational matrix and, then, clustered normally using a relational clustering algorithm. Moreover, there is a well-established approach for cluster ensemble that consists of the mapping of a set of partitions into a relational matrix (named co-association matrix) followed by the application of a relational algorithm to this matrix [5].

When a partitional clustering algorithm [6], [7] is applied to a data set – no matter whether it is relational or not – the result is a partition $C = \{C_1, C_2, \ldots, C_k\}$ of the data into k clusters, such that:

$$\begin{aligned} &\bigcup_{i=1}^{k} C_i = O \\ &C_i \neq \emptyset \qquad \text{for all } i \in \{1, \ldots, k\} \\ &C_i \cap C_j = \emptyset \quad \text{for all } i, j \in \{1, \ldots, k\} : \ i \neq j \end{aligned} \tag{1}$$

where C_i stands for the ith cluster. The estimation of an appropriate value for k is a key problem in practical clustering applications. A widely known and simple approach to tackle this problem consists of using a systematic (repetitive) procedure to run a given clustering algorithm multiple times for different numbers of clusters and, then, selecting the best partition according to a clustering validity criterion. Another major approach is to use some sort of meta-heuristic, such as evolutionary algorithms, specially designed to perform a wiser, guided search through the space of candidate partitions with variable k [8], [9]. In particular, the Evolutionary Algorithm for Clustering (EAC) proposed in [10] was designed to evolve partitions with variable k by eliminating, splitting, and merging clusters that are refined by the classic k-means algorithm. EAC has been shown – from a statistical perspective – to be more efficient than systematic (repetitive) approaches based on multiple runs of k-means when the number of clusters in a data set is unknown [10], [11]. In [12], the authors showed that the computational efficiency of EAC could be further improved with the use of guided mutation operators with self-adjusting application rates, among other additional features. The incorporation of those features gave rise to the Fast Evolutionary Algorithm for Clustering (F-EAC), which is of foremost interest in the present paper.

In this paper, we propose two modified versions of F-EAC for clustering of relational data. These modified versions are derived by rewriting the mutation operators, the fitness function, and the local search procedure of the original F-EAC so that they all become relational. The computational efficiency of the proposed relational variants of F-EAC is experimentally assessed in eight data sets and the results are statistically compared against those obtained by repetitive procedures.

The remainder of this paper is organized as follows. Section II describes the novel evolutionary algorithms for relational clustering. Section III presents an extensive collection of experiments in which the proposed algorithms are tested against traditional repetitive procedures. Finally, Section IV addresses the conclusions.

II. Fast Evolutionary Algorithm for Relational Clustering (F-EARC)

Two relational variants of the original F-EAC are introduced in this section. They differ from one another by the use of distinct local search procedures: Basic k-Medoids (BKM)[1] and Relational Hard c-Means (RHCM) [2]. The variant that uses BKM will be hereafter named F-EARC-BKM, while the one that uses RHCM will be termed F-EARC-RHCM. In order to refer to the relational variants in general, no matter the local search procedure in use, the acronym F-EARC will be adopted. Algorithm 1 depicts the general outline of F-EARC, whose steps are described in the following sections. The parameters of the algorithm are: (i) the size of the population, m; and (ii) the maximum number of iterations, t, that can be performed in a single application of the local search procedure (BKM or RHCM $-$ Step 3).

Algorithm 1 F-EARC.

```
1: Randomly initialize a population of $m$ genotypes encoding clustering partitions (Sections II-A and II-B);
2: while Stopping criterion is not satisfied do
3:     Apply BKM or RHCM to every individual of the population (Section II-C);
4:     Evaluate the fitness of every individual by means of a clustering validity criterion suitable for relational clustering (e.g., the Silhouette index [4]);
5:     Apply rank-based linearization [13] to the fitness values;
6:     Select individuals (Section II-E);
7:     Apply the relational mutation operators MO1 e MO2 (Section II-D) to the selected individuals, according to a guided application policy (Section II-E);
8:     Replace the old population with the new one;
9: end while
```

[1]BKM is essentially the classic k-means algorithm with centroids substituted by representative objects (medoids) that are computed so as to minimize the sum of distances between them and the objects of the corresponding clusters; i.e., the object selected as medoid of a cluster C_i is such that $\tilde{o}_i = \arg\min_{\tilde{o} \in C_i} \sum_{o \in C_i} d(o, \tilde{o})$, where d is a distance measure.

A. Genotype encoding

The genotypes of F-EARC encode cluster prototypes. In the case of F-EARC-BKM, the prototypes are medoids, whereas in the case of F-EARC-RHCM, the prototypes are relational centroids (defined further in Section II-B). Since the mutation operators act directly on the logic level of the individuals (phenotype), no matter the peculiarities adopted at the implementation level, the detailing of the genotype encoding is unnecessary; it suffices to notice that the prototypes must be stored in the genotypes. From the prototypes, the corresponding clustering partitions are restored simply by assigning the data objects to the nearest prototypes.

B. Settings and initial population

In this work, the prototypes of the initial population are generated according to a typical approach [14], [15], [16], [17], [18]. For each individual, a value k is randomly drawn from the interval $[2, \sqrt{n}]$ (whose upper limit is a commonly used rule of thumb [19], [20]), then k objects $\tilde{o}_1, \ldots, \tilde{o}_k$ are randomly drawn from the data set, without reposition. In the case of F-EARC-BKM, these objects are the prototypes (medoids) to be stored in the respective genotype. In the case of F-EARC-RHCM, these objects are used to derive the prototypes (relational centroids). Specifically, the drawn objects are used to obtain a partition $C = \{C_1, \ldots, C_k\}$ that satisfies the constraints in (1), according to the following rule:

$$o_l \in C_i \text{ iff } i = \arg\min_{j} d(o_l, \tilde{o}_j), \tag{2}$$

where $l = 1, \cdots, n$, $i = 1, \cdots, k$, and $d(\cdot, \cdot)$ stands for the dissimilarity (distance) between two objects; ties are arbitrarily broken. Once partition C has been recovered, k relational centroids (cluster prototypes for F-EARC-RHCM) can be computed, as follows [2]:

$$v_i = (u_{i1}, u_{i2}, \ldots, u_{in})^T / \sum_{l=1}^{n} u_{il}, \tag{3}$$

where $u_{il} = 1$ if the lth data object belongs to the ith cluster ($o_l \in C_i$) and $u_{il} = 0$ otherwise ($o_l \notin C_i$). It is worth noticing that v_i, which is the relational centroid of cluster C_i, is an n-dimensional vector, where n is the number of objects in the data set. From these relational centroids, the distances between objects and centroids, which are required by the RHCM algorithm, can be computed using the (Euclidean) relational matrix of the data to be clustered [2].

C. Local search

F-EARC, like all the other variants of the Evolutionary Algorithm for Clustering (EAC) [10], [12], [21], [11], has a local search procedure by which a clustering algorithm is applied to the individuals so as to refine the corresponding partitions, thus speeding up the global search performed by the evolutionary operators. F-EARC-BKM and F-EARC-RHCM use BKM and RHCM as their local search procedures, respectively. These procedures, in turn, are endowed with two stopping criteria: (i) all the cluster prototypes have converged; or (ii) a maximum number of iterations (t) have been completed. Since the evolutionary search performed by F-EARC provides a cumulative refinement of the solutions in the course of generations, it is expected that a few iterations of BKM or RHCM will be required to make F-EARC both fast and effective. Such a sort of hypothesis has been reinforced in [22] by means of an in-depth experimental analysis of the effect of the number of local search iterations performed by an EAC-like algorithm. These experiments and previous experience indicate that this family of algorithms is quite robust to the choice of this parameter and also suggest that values for t not greater than $t = 5$ will usually suffice to provide accurate results with reduced computational requirements. Indeed, BKM and RHCM are relational variants of the classic *k-means* algorithm. Empirical evidence suggests that five or fewer iterations are usually enough for *k-means* to find satisfactory solutions [23], even when running in a standalone environment (without the support of an evolutionary guidance, for instance).

D. Mutation Operators

F-EARC is equipped with two mutation operators capable of dealing with relational data sets (Step 7 of Algorithm 1). The first operator (MO1) is responsible for the removal of a subset of selected clusters from a given individual and, accordingly, for reducing the number of clusters of the corresponding partition. MO1 acts by removing $-$ from the respective individual $-$ the prototypes of the selected clusters. At the subsequent generation, when the mutated individual is subject to the local search procedure (Step 3 of Algorithm 1), the objects from the removed clusters are naturally reallocated to the nearest remaining clusters. The selection of clusters to be removed takes place probabilistically, taking into account a measure of quality of the individual clusters. Specifically, higher quality clusters are more likely to be preserved than lower quality ones. To that end, the value $-S(C_i)$ is assigned to cluster C_i, for all $i \in \{1, \ldots, k\}$, where $S(\cdot)$ is the Silhouette index [4] computed exclusively with respect to those objects belonging to cluster C_i. These values undergo a rank-based linearization [13] and are then used to select those clusters to be removed. Particularly, the well-known roulette wheel strategy [24] (without reposition) is adopted. MO1's steps are presented in Algorithm 2.

The second mutation operator (MO2) is responsible for splitting a subset of clusters selected from a given individual to be mutated and, accordingly, increasing the number of clusters of the corresponding partition. MO2 acts by replacing the prototype of each selected cluster with a pair of new ones. Let C_s be a cluster selected for splitting. The prototype of C_s is removed and two objects of C_s ($o_s^{'}$ and $o_s^{''}$) are chosen. The first object, $o_s^{'}$, is randomly chosen from C_s. The second object, $o_s^{''}$, is the object of C_s farthest from $o_s^{'}$. In the case of F-EARC-BKM, $o_s^{'}$ and $o_s^{''}$ are stored as a new pair of prototypes (medoids), in place of the one that has been removed. In the case of F-EARC-RHCM, it is necessary to recover two clusters from $o_s^{'}$ and $o_s^{''}$ before obtaining the new pair of prototypes (as relational centroids). To that end, the objects of C_s closer to $o_s^{'}$ form cluster $C_s^{'}$, whereas those objects of C_s closer to $o_s^{''}$ form cluster $C_s^{''}$. The corresponding pair of relational centroids (prototypes) can thus be computed using Equation (3). The selection of the clusters to be split follows precisely the same probabilistic procedure used by the first mutation operator (MO1). For the sake of compactness, a step-by-step description of MO2 will be omitted here.

Algorithm 2 Mutation Operator 1 (MO1).

```
1:  Let $g$ be a genotype to be mutated and $k$ be the number
    of clusters in the data partition $C$ encoded into $g$;
2:  if $k > 2$ then
3:      for $i = 1, \ldots, k$ do
4:          Calculate the value of the Silhouette index, $S(C_i)$,
            associated with cluster $C_i$;
5:      end for
6:      Apply rank-based linearization to the values
        $-S(C_1), \ldots, -S(C_k)$ so that lower values are
        assigned to better clusters;
7:      Randomly draw a number $h \in \{1, \ldots, k-2\}$;
8:      for $i = 1, \ldots, h$ do
9:          Draw a cluster $C_s$ from $C$ using the roulette
            wheel strategy, without reposition, according to the
            linearized values assigned to clusters $C_1, \cdots, C_k$ in
            Step 6;
10:         Update $g$ by removing the prototype associated
            with cluster $C_s$;
11:     end for
12: end if
```

E. Selection and guided application of the mutation operators

F-EARC, like EAC, uses by default the well-known roulette wheel strategy as well as unitary elitism as its selection mechanisms[2] [24]. In this case, the fittest individual of the current population (of size m) is preserved to be directly conveyed to the next generation (unitary elitism). The remaining $m - 1$ individuals that will form the next generation are drawn from the current population by means of the roulette wheel strategy (proportional selection), with

[2]Any other alternative selection operator can be employed, if desired.

reposition. After selection, the mutation operators MO1 and MO2 are applied to these $m-1$ selected individuals.

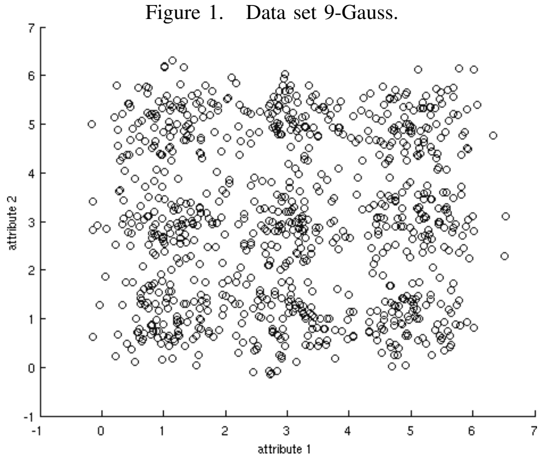
In the present paper, the application of the mutation operators follows a novel (individual-oriented) guidance policy: if the fitness of the ith individual has increased from the last generation to the current one, then apply once again the same mutation operator that was applied to that individual in the last generation. Contrarily, if the fitness of the ith individual has decreased, then change the mutation operator. Finally, if the ith individual has not undergone mutation in the last generation (for it has been preserved by the elitist strategy or in case the current generation refers to the initial population), then randomly choose between MO1 or MO2 with equal probabilities. The rationale behind this approach is very simple and intuitive: keep applying operator MOi as long as improvements are observed; otherwise, change the mutation operator. This idea is better justified when the search for the most natural number of clusters is considered. Suppose, for example, that the ith individual encodes a partition with k clusters, where k is much higher than k^* (the most natural number of clusters in the data set). It is reasonable to conjecture that the fitness of this individual tends to increase if the number of clusters it encodes, k, is reduced. From this perspective, it is legitimate to try keeping the application of MO1 to that individual as long as improvements in its fitness are observed. This is likely to happen until the number of clusters k becomes lower than k^*.

It is worth remarking that the above-mentioned approach for the guided application of the mutation operators is different from that which was originally used by F-EAC [12]. In Section III-A, experiments involving a number of data sets provide empirical evidence that this novel approach may outperform the original one, besides being simpler.

III. Experiments

Six artificial and two real data sets are considered here. Specifically, five artificial data sets, hereafter referred to as Bio1, Bio2, Bio3, Bio4, and Bio5, consist of synthetic gene-expression data with error distributions derived from real data [25]. These data sets are available in [26] (those without repeated measurements and with low noise levels). They have 400 objects (genes) each, almost equally distributed between six clusters, and are described by 20 attributes (measures). The sixth artificial data set is 9-Gauss (Figure 1) [11][3]. 9-Gauss has 9 overlapping clusters, each of which formed by 100 objects generated from bidimensional Gaussian distributions with equal variances. One of the real data sets is the yeast galactose data (Yeast for short) [27], [25]. As in [25], we use a subset of 205 objects (genes), whose expression patterns reflect 4 functional categories (clusters). The second real data set considered here is the

Figure 1. Data set 9-Gauss.



well-known Original Breast Cancer Wisconsin data (Breast for short) [28]. For each data set, a relational data matrix R is computed using the Euclidian distance (normalized w.r.t. the number of present attribute values, since Breast has 16 objects with missing values). This matrix is the only information to be provided as input to F-EARC.

Let us recall from Section II that the parameters of F-EARC are the population size, m, and the maximum number of local search iterations, t. The latter is set here as $t = 5$, for the reasons already discussed in Section II-C. In what concerns the population size, this parameter is not critical for most evolutionary algorithms. Roughly speaking, large populations tend to cause the evolutionary search to converge to a good solution in a few generations (each of them computationally costly due to the large population size). Small populations, in opposite, will likely make the algorithm converge after a higher number of generations, which however tend to be individually less computationally costly, since only a few individuals will be processed. Particularly, EAC-like algorithms have shown to be quite robust to the choice of m [12], [11]. Actually, they have shown to be effective even when evolving a very small population of solutions, such as $m = 4$, over real data [10]. In spite of this, empirical evidence suggests that values around $m = 10$ provide a better trade-off between required accuracy and the overall processing time needed to achieve it [11]. For this reason, a population of $m = 10$ individuals is adopted here. For the sake of a more fair comparison against systematic (repetitive) methods, to be discussed in the sequel, the numbers of clusters encoded into the individuals of the population are forced to stay within the range $[2, \sqrt{n}]$, though this is not necessary in real applications.

Two widespread repetitive methods that automatically estimate the number of clusters in data are considered for comparison purposes. The first one, here referred to as Ordered Multiple Runs (OMR), gets two positive integers, n_p and t, an interval $[k_{min}, k_{max}]$, a clustering algorithm E, and

[3]Available at www.icmc.usp.br/~campello/Sub_Pages/JH.htm.

Table I
AVERAGE RUNNING TIMES (S) FOR F-EARC.

		data sets							
	app	Bio1	Bio2	Bio3	Bio4	Bio5	Yeast	9-Gauss	Breast
BKM	o	10.6	5.7	4.6	11.5	13.9	1.8	88.6	9.1
	n	9.0	5.9	5.2	11.4	10.4	1.8	66.2	7.5
RHCM	o	12.1	7.3	10.1	18.6	17.4	1.6	61.6	23.0
	n	14.1	7.1	9.8	16.5	12.3	1.8	57.1	19.9

Table II
AVERAGE RUNNING TIMES (S): BKM.

	data sets							
Algorithm	Bio1	Bio2	Bio3	Bio4	Bio5	Yeast	9-Gauss	Breast
OMR (5)	6.3	6.3	6.4	6.3	6.4	2.4	26.4	15.8
OMR (10)	12.6	12.6	12.9	12.7	12.8	4.8	52.7	31.5
OMR (15)	18.9	18.9	19.3	19.1	19.3	7.2	79.1	47.3
OMR (20)	25.2	25.1	25.7	25.4	25.6	9.6	105.6	63.0
MR	8.1	4.9	-	-	118.2	5.1	125.2	3.3
F-EARC	6.2	5.9	7.8	12.8	11.0	1.7	75.3	7.7

Table III
AVERAGE RUNNING TIMES (S): RHCM.

	data sets							
Algorithm	Bio1	Bio2	Bio3	Bio4	Bio5	Yeast	9-Gauss	Breast
OMR (5)	18.0	18.0	18.2	17.8	18.0	5.2	129.1	68.9
OMR (10)	35.8	36.0	36.4	35.5	35.9	10.3	258.6	138.0
OMR (15)	53.8	54.0	54.6	53.5	53.8	15.5	388.7	207.0
OMR (20)	71.7	72.0	72.8	71.1	71.7	20.7	518.2	276.0
MR	19.1	11.7	-	-	311.2	2.3	245.1	25.6
F-EARC	11.3	8.4	8.8	26.4	12.8	1.6	67.2	19.0

a relative clustering validity criterion V (e.g., the Silhouette criterion) as inputs. OMR executes E, for a maximum of t iterations, n_p times for each $k \in [k_{min}, k_{max}]$. The partition with the best value of V is returned as the output. The second method, here referred to as Multiple Runs (MR), gets t, $[k_{min}, k_{max}]$, an algorithm E, a validity criterion V, and a reference (desired) validity value S^* for V as inputs. MR repeatedly draws a value of k (randomly) from $[k_{min}, k_{max}]$ and runs E for at most t iterations, until a partition with value of V equal to or better than S^* is found. In order to specify the embedded clustering algorithms, E, we refer here to the names OMR-BKM, OMR-RHCM, MR-BKM, and MR-RHCM, which are self-explainable.

A. Analysis of the application method of the mutation operators

In a preliminary experiment, F-EARC was executed 50 times for each data set using two different approaches for the guided application of the mutation operators: the novel approach described in Section II-E and the one originally employed by F-EAC [12]. In order to set up a stopping criterion for F-EARC, OMR was run with $n_p = 100$, $t = \infty$, $[k_{min}, k_{max}] = [2, \sqrt{n}]$, $E =$ BKM (RHCM), and $V =$ Silhouette. The best Silhouette value achieved, S^*_{BKM} (S^*_{RHCM}), was taken as a stopping criterion for F-EARC-BKM (F-EARC-RHCM); i.e., F-EARC stopped whenever a partition with Silhouette equal to or larger than S^*_{BKM} (S^*_{RHCM}) was found. Table I displays the average running times. In this table, "app = n" refers to the **n**ovel **app**roach described in Section II-E, whereas "app = o" refers to the one employed by the **o**riginal F-EAC. Table I shows that the overall performance of the original application method is worse. In only 4 out of 16 experiments the original method showed slightly better average times. For this reason, the novel, simpler method described in Section II-E was adopted in the subsequent experiments.

B. F-EARC versus repetitive approaches

For each data set, OMR-BKM (OMR-RHCM) was run 30 times with $t = 5$, $V =$ Silhouette, $[k_{min}, k_{max}] = [2, \sqrt{n}]$, and $n_p = 5, 10, 15,$ and 20. The best Silhouette value S^*_{BKM} (S^*_{RHCM}) out of these $30 \times 4 = 120$ runs was taken as the reference value for MR-BKM (MR-RHCM), i.e., $S^* = S^*_{\text{BKM}}$ ($S^* = S^*_{\text{RHCM}}$). MR-BKM (MR-RHCM) was then run 30 times for each data set, using the same settings for t, V, and $[k_{min}, k_{max}]$. Finally, F-EARC-BKM (F-EARC-RHCM) was also run 30 times for each data set, seeking the same reference value S^*_{BKM} (S^*_{RHCM}). Tables II and III summarize the results. Symbol "-" indicates that a single run of MR exceeded 30 times the average running time of OMR for the respective data set. In these cases, we decided to halt the corresponding experiments for the conclusions had already become evident.

It can be seen from Tables II and III that F-EARC outperformed MR in 14 out of 16 experiments (8 data sets $\times$ 2 clustering algorithms). Particularly, MR's performance degraded in a noticeable manner for data sets Bio3, Bio4, and Bio5. We conjecture that these data sets are more complex than Bio1 and Bio2 from the clustering perspective. This conjecture stems from the fact that OMR-BKM (OMR-RHCM) achieved maximum Silhouette values S^*_{BKM} (S^*_{RHCM}) much less frequently when running (30 times) over Bio3, Bio4, and Bio5 than when running over Bio1 and Bio2. Likewise, F-EARC also performed significantly better than MR for data set 9-Gauss. We believe that this is due to the presence of highly overlapped clusters in 9-Gauss, which makes the corresponding clustering problem more difficult. These results suggest that the informed search strategy employed by F-EARC is more suited to handle more complex data sets.

Tables II and III also show that F-EARC performed better than OMR for $n_p = 5$ in 11 out of 16 experiments, and much more prominently for $n_p > 5$. These results become even more expressive when one recalls that: (i) the reference value S^*_{BKM} (S^*_{RHCM}) for the stopping criterion of F-EARC was taken as the highest one attained by OMR-BKM (OMR-RHCM) among all its runs; and (ii) not all runs of OMR-BKM (OMR-RHCM) achieved a partition with such a maximum quality. This can be seen from Tables

Table IV
AVERAGE SILHOUETTE VALUES: BKM.

	data sets							
Algorithm	Bio1	Bio2	Bio3	Bio4	Bio5	Yeast	9-Gauss	Breast
OMR (5)	0.64	0.68	0.71	0.69	0.64	0.50	0.48	0.60
OMR (10)	0.64	0.69	0.72	0.69	0.64	0.50	0.48	0.60
OMR (15)	0.64	0.69	0.72	0.69	0.64	0.51	0.49	0.60
OMR (20)	0.64	0.69	0.72	0.69	0.64	0.51	0.49	0.60
MR	0.65	0.69	-	-	0.64	0.51	0.49	0.60
F-EARC	0.65	0.69	0.73	0.70	0.64	0.51	0.49	0.60

Table V
AVERAGE SILHOUETTE VALUES: RHCM.

	data sets							
Algorithm	Bio1	Bio2	Bio3	Bio4	Bio5	Yeast	9-Gauss	Breast
OMR (5)	0.64	0.69	0.72	0.69	0.64	0.51	0.48	0.60
OMR (10)	0.64	0.69	0.72	0.69	0.64	0.51	0.48	0.60
OMR (15)	0.64	0.69	0.72	0.69	0.64	0.51	0.49	0.60
OMR (20)	0.65	0.69	0.72	0.69	0.64	0.51	0.49	0.60
MR	0.65	0.69	-	-	0.64	0.51	0.49	0.60
F-EARC	0.65	0.69	0.73	0.70	0.64	0.51	0.49	0.60

IV and V, which display the mean values of the Silhouette criterion. Since the highest Silhouette value obtained by OMR is a lower bound for F-EARC, F-EARC performed equal to or better than OMR across all experiments. This is also observed in most columns of Tables VI and VII, which display the mean values of the Jaccard external index [7], [6].

To give better confidence to our conclusions, we applied the two-sided Wilcoxon rank-sum statistical test ($\alpha = 5\%$) [29] to the results. Tables VIII and IX show the results of individual tests between F-EARC and the repetitive algorithms. Symbol $\triangle$ means that the null hypothesis is rejected and F-EARC has a lower ranking sum (F-EARC is faster). Symbol $\bigtriangledown$ means that the null hypothesis is rejected and the corresponding repetitive algorithm has a lower ranking sum (F-EARC is slower). Symbol $\circ$ means that there is no statistical evidence to reject the null hypothesis. It is important to notice that F-EARC has only been considered slower (with statistical significance) in the following scenarios: (i) when compared to OMR with $n_p = 5$ in four data sets; and (ii) when compared to MR in the Breast data set. In the first scenario, we did not expect F-EARC to perform better than OMR (even though that happened for most data sets), inasmuch as five repetitions ($n_p = 5$) for each k is usually a very optimistic value for OMR-like approaches. In what concerns the performance of F-EARC against MR for the Breast data, we once again conjecture that the reason is the relative simplicity of this data set from the clustering perspective, in view of the fact that MR was at least five times faster than OMR with $n_p = 5$ for these data.

In summary, 62 out of 80 experiments provided statistical evidence in favor of F-EARC as a faster algorithm.

Table VI
AVERAGE JACCARD VALUES: BKM.

	data sets							
Algorithm	Bio1	Bio2	Bio3	Bio4	Bio5	Yeast	9-Gauss	Breast
OMR (5)	0.93	0.71	0.58	0.95	0.81	0.82	0.76	0.86
OMR (10)	0.97	0.75	0.59	0.98	0.74	0.83	0.78	0.86
OMR (15)	0.99	0.75	0.59	1.00	0.75	0.84	0.79	0.86
OMR (20)	0.98	0.75	0.59	1.00	0.75	0.84	0.80	0.86
MR	1.00	0.75	-	-	0.74	0.84	0.80	0.86
F-EARC	1.00	0.75	0.59	0.99	0.74	0.84	0.81	0.86

Table VII
AVERAGE JACCARD VALUES: RHCM.

	data sets							
Algorithm	Bio1	Bio2	Bio3	Bio4	Bio5	Yeast	9-Gauss	Breast
OMR (5)	0.90	0.74	0.59	0.96	0.78	0.84	0.76	0.87
OMR (10)	0.98	0.75	0.59	1.00	0.75	0.84	0.78	0.87
OMR (15)	0.99	0.75	0.59	1.00	0.75	0.84	0.79	0.87
OMR (20)	1.00	0.75	0.59	1.00	0.75	0.84	0.79	0.87
MR	1.00	0.75	-	-	0.74	0.84	0.79	0.87
F-EARC	1.00	0.75	0.59	0.99	0.74	0.84	0.79	0.87

Table VIII
WILCOXON RANK-SUM TEST: BKM.

	F-EARC-BKM on data sets							
Algorithm	Bio1	Bio2	Bio3	Bio4	Bio5	Yeast	9-Gauss	Breast
OMR (5)	$\circ$	$\triangle$	$\circ$	$\bigtriangledown$	$\bigtriangledown$	$\triangle$	$\bigtriangledown$	$\triangle$
OMR (10)	$\triangle$	$\triangle$	$\triangle$	$\circ$	$\triangle$	$\triangle$	$\circ$	$\triangle$
OMR (15)	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\circ$	$\triangle$
OMR (20)	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\circ$	$\triangle$
MR	$\circ$	$\circ$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\circ$	$\bigtriangledown$

Table IX
WILCOXON RANK-SUM TEST: RHCM.

	F-EARC-RHCM on data sets							
Algorithm	Bio1	Bio2	Bio3	Bio4	Bio5	Yeast	9-Gauss	Breast
OMR (5)	$\triangle$	$\triangle$	$\triangle$	$\bigtriangledown$	$\triangle$	$\triangle$	$\triangle$	$\triangle$
OMR (10)	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$
OMR (15)	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$
OMR (20)	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$	$\triangle$
MR	$\circ$	$\circ$	$\triangle$	$\triangle$	$\triangle$	$\circ$	$\triangle$	$\circ$

IV. CONCLUSIONS

We have addressed the use of evolutionary algorithms to tackle the problem of clustering relational data sets when the number of clusters is unknown. Particularly, we have proposed two relational versions of a Fast Evolutionary Algorithm for Clustering (F-EAC) that are endowed with evolutionary operators capable of dealing with relational data and variable numbers of clusters. The proposed evolutionary algorithms for relational clustering have been statistically assessed in eight data sets against two traditional repetitive approaches that can also be used to automatically estimate the number of clusters in relational data. The proposed algorithm outperformed the repetitive methods in terms of accuracy and processing time as well. For this reason, we believe that the proposed algorithms are promising tools to tackle relational clustering problems when the number of clusters is not known in advance.

Acknowledgments

The authors thank CNPq and FAPESP for the financial support and Murilo C. Naldi for his constructive suggestions.

References

[1] R. J. Hathaway and J. C. Bezdek, "Nerf c-means: Non-euclidean relational fuzzy clustering," *Pattern Recognition*, vol. 27, no. 3, pp. 429–437, March 1994.

[2] R. J. Hathaway, J. W. Davenport, and J. C. Bezdek, "Relational duals of the c-means clustering algorithms," *Pattern Recognition*, vol. 22, no. 2, pp. 205–212, 1989.

[3] R. Dave and S. Sen, "Robust fuzzy clustering of relational data," *IEEE Trans. Fuzzy Systems*, vol. 10, no. 6, pp. 713–727, Dec 2002.

[4] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data: an introduction to cluster analysis*. Wiley, 1990.

[5] A. L. N. Fred and A. K. Jain, "Combining multiple clusterings using evidence accumulation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 6, pp. 835–850, 2005.

[6] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1988.

[7] B. S. Everitt, S. Landau, and M. Leese, *Cluster Analysis*. Arnold Publishers, May 2001.

[8] E. Falkenauer, *Genetic Algorithms and Grouping Problems*. New York City, NY, USA: John Wiley & Sons, Inc., 1998.

[9] E. R. Hruschka, R. J. G. B. Campello, A. A. Freitas, and A. C. P. L. F. de Carvalho, "A survey of evolutionary algorithms for clustering," *IEEE Trans. Systems, Man and Cybernetics, Part C*, vol. 39, no. 2, pp. 133–155, 2009.

[10] E. R. Hruschka, R. J. G. B. Campello, and L. N. de Castro, "Evolving clusters in gene-expression data," *Information Sciences*, vol. 176, pp. 1898–1927, 2006.

[11] R. J. G. B. Campello, E. R. Hruschka, and V. S. Alves, "On the efficiency of evolutionary fuzzy clustering," *Journal of Heuristics*, vol. 15, no. 1, pp. 43–75, 2009.

[12] V. S. Alves, R. J. G. B. Campello, and E. R. Hruschka, "Towards a fast evolutionary algorithm for clustering," in *CEC 2006. IEEE Congress on Evolutionary Computation*, 2006, pp. 1776–1783.

[13] L. Davis, *Handbook of Genetic Algorithms*. International Thomson Computer Press, 1996.

[14] L. I. Kuncheva and J. C. Bezdek, "Selection of cluster prototypes from data by a genetic algorithm," in *Proc. 5th European Congress on Intelligent Techniques and Soft Computing*, Aachen, Alemanha, 1997, pp. 1683–1688.

[15] S. Bandyopadhyay and U. Maulik, "An evolutionary technique based on k-means algorithm for optimal clustering in Rn," *Information Sciences*, vol. 146, no. 1-4, pp. 221 – 237, 2002.

[16] ——, "Genetic clustering for automatic evolution of clusters and application to image classification," *Pattern Recognition*, vol. 35, no. 6, pp. 1197–1208, June 2002.

[17] C. B. Lucasius, A. D. Dane, and G. Kateman, "On k-medoid clustering of large data sets with the aid of a genetic algorithm: background, feasibility and comparison," *Analytica Chimica Acta*, vol. 287, pp. 647–669, 1993.

[18] V. Estivill-Castro and A. T. Murray, "Spatial clustering for data mining with genetic algorithms," University of Queensland, Australia, Tech. Rep., 1997.

[19] N. R. Pal and J. C. Bezdek, "On cluster validity for the fuzzy c-means model," *IEEE Trans. Fuzzy Systems*, vol. 3, no. 3, pp. 370–379, 1995.

[20] M. K. Pakhira, S. Bandyopadhyay, and U. Maulik, "A study of some fuzzy cluster validity indices, genetic clustering and application to pixel classification," *Fuzzy Sets and Systems*, vol. 155, no. 2, pp. 191 – 214, 2005.

[21] E. R. Hruschka, L. de Castro, and R. J. G. B. Campello, *Clustering Gene-Expression Data: A Hybrid Approach that Iterates Between k-Means and Evolutionary Search*. Springer Berlin / Heidelberg, 2007 - available at http://www.icmc.usp.br/~campello/Sub_Pages/Selected_Publications.htm, pp. 313–335.

[22] D. Horta, M. C. Naldi, R. J. G. B. Campello, E. R. Hruschka, and A. C. P. L. F. Carvalho, "Evolutionary fuzzy clustering: An overview and efficiency issues," in *Foundations of Computational Intelligence*, A. Abraham, A.-E. Hassanien, and A. C. P. L. F. Carvalho, Eds. Springer-Verlag, 2009 - available at http://www.icmc.usp.br/~campello/Sub_Pages/Selected_Publications.htm, vol. 4, pp. 167–195.

[23] M. R. Anderberg, *Cluster Analysis for Applications*. Academic Press Inc., 1973.

[24] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1989.

[25] K. Yeung, M. Medvedovic, and R. Bumgarner, "Clustering gene-expression data with repeated measurements," *Genome Biology*, vol. 4, 2003.

[26] ——, 2003, http://expression.microslu.washington.edu/expression/kayee/cluster2003/yeunggb2003.html.

[27] A. E. Raftery, K. Y. Yeung, K. Y. Yeung, C. Fraley, C. Fraley, A. Murua, A. Murua, W. L. Ruzzo, and W. L. Ruzzo, "Model-based clustering and data transformation for gene expression data," *Bioinformatics*, vol. 17, pp. 977–987, 2001.

[28] A. Asuncion and D. J. Newman, "UCI machine learning repository," 2007, university of California, Irvine, School of Information and Computer Sciences – www.ics.uci.edu/∼mlearn/MLRepository.html.

[29] M. Hollander and D. A. Wolfe, "Nonparametric statistical methods," Canada, 1999.