An Innovative Approach to Genetic Programming–based Clustering

I. De Falco[1], E. Tarantino[1], A. Della Cioppa[2] and F. Fontanella[3]

[1] Institute of High Performance Computing and Networking
ICAR - CNR Via P. Castellino, 111 80131 Naples – Italy
[2] Department of Computer Science and Electrical Engineering
University of Salerno 84084 Fisciano (SA) – Italy
[3] Department of Information Engineering and Systems
University of Naples, Via Claudio, 21 80125 Naples – Italy

Abstract. Most of the classical clustering algorithms are strongly dependent on, and sensitive to, parameters such as number of expected clusters and resolution level. To overcome this drawback, in this paper a Genetic Programming framework, capable of performing an automatic data clustering, is presented. Moreover, a novel way of representing clusters which provides intelligible information on patterns is introduced together with an innovative clustering process. The effectiveness of the implemented partitioning system is estimated on a medical domain by means of evaluation indices.

1 Introduction

Clustering is the part of data mining whose task consists in grouping a set of similar objects based on some measures of goodness that differ according to application [1, 2]. Differently from the supervised learning in which, given a collection of labeled training examples used to learn the descriptions of classes, the problem is to label a newly encountered pattern, in clustering the problem is to group a given collection of unknown patterns into meaningful clusters with no assumptions about the relationships.

The clustering aim is to develop methods and tools for analyzing large data sets and for searching for unexpected relationships in the data. A variety of clustering algorithms have been developed to face the problem [3]. Most of these algorithms require user inputs of several parameters like the number of clusters and the average dimensionality of the cluster [4–6], which are not only difficult to determine but are also not practical for real–world data sets. In the hierarchical algorithms a human user evaluates a posteriori the resolution level, and thus the cluster number, by which he wants to partition the assigned data set. In other cases, different criteria have been introduced to find the optimal value of the cluster number [7–9]. Hence, the output of these clustering algorithms is very sensitive to user expertise.

To overcome these drawbacks, evolutionary techniques have been applied to cluster analysis [10–13]. However pure evolutionary techniques are normally

considered inefficient due to high computational costs [10] and thus the usage of combined strategies has been attempted [9, 14]. In this paper the effectiveness of a Genetic Programming (GP) [15] framework to perform automatic data clustering without providing any kind of user knowledge is investigated. The hope is that GP, based only on information implicitly contained in the data set, allows to find an efficient partitioning with interesting correlations among patterns. To this aim an innovative way of representing clusters, based on logical formulas, is introduced. Furthermore a new clustering procedure tied to this representation is proposed. A solution is found by maximizing intra–cluster homogeneity and inter–cluster separability of the clustering. This evolutionary system is applied to clustering in a medical domain.

The paper is organized as follows: in Sect. 2 a formalization of data clustering is reported, while in Sect. 3 our GP–based clustering system is outlined together with its implementation details. In Sect. 4 the test problem and some evaluation indices are described. The performance of our system is discussed in Sect. 5. The last section illustrates our final remarks and future work.

2 Data Clustering

2.1 Definitions and Notations

Each element of the database is said *pattern* and is represented by $\mathbf{X} = (x_1, \ldots, x_\ell)$ with $\mathbf{X} \in \mathbf{S}$, where $\mathbf{S}$ is the universe of all possible elements with ℓ attributes and x_j denotes the j–th attribute of the pattern. A *pattern set* $\mathcal{X}$ with cardinality n is denoted with $\mathcal{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_n)$ with, in general, $\mathcal{X} \subseteq \mathbf{S}$.

Distance between Patterns. Definition of distance between patterns depends on the type of attributes. For binary attributes the Hamming distance is used while for numerical ones the linear distance $\delta(x_j, y_j) = |x_j - y_j|$ is considered.

Once defined the range for the j-th attribute as: $R_j \equiv \delta(\max_{i=1,\ldots,n} x_{ij}, \min_{i=1,\ldots,n} x_{ij})$ where x_{ij} indicates the j–th attribute of the i–th pattern, the distance between patterns chosen is the Manhattan one:

$$d(\mathbf{X}, \mathbf{Y}) = \frac{1}{\ell} \cdot \sum_{j=1}^{\ell} \left(\frac{1}{R_j} \cdot \delta(x_j, y_j) \right) \tag{1}$$

This distance is normalized with respect to the range R_j and to the number of attributes, so as to have $0 \leq d(\mathbf{X}, \mathbf{Y}) \leq 1 \ \forall \ \mathbf{X}, \mathbf{Y} \in \mathcal{X}$.

Cluster Representative and Clustering. Several ways are possible to represent a cluster $\mathbf{C}_k$. In the enumerative method for each cluster the data set elements which belong to it are listed. In others a cluster is represented by listing for each attribute the maximum and the minimum values computed on cluster members. In many algorithms a representative for any cluster $\mathbf{C}_k$ is used,

i.e. an element of set **S** which can whether or not belong to the cluster. However, the most used representative is the centroid $\mathbf{P}_k$ defined as the average point for patterns belonging to $\mathbf{C}_k$. In this case, a clustering **CL** with cardinality $\mathcal{N}$ can be denoted by the list of its cluster representatives: $\mathbf{CL} \equiv \{\mathbf{P}_1, \ldots, \mathbf{P}_\mathcal{N}\}$.

2.2 Scoring Function, Homogeneity and Separability

To quantify the clustering quality with respect to data set taken into account, a scoring function f_s, expressed in terms of homogeneity $\mathcal{H}$ and separability $\mathcal{S}$ for the clustering **CL**, is considered. Thus a generic score function is:

$$f_s = f(\mathcal{H}(\mathbf{CL}), \mathcal{S}(\mathbf{CL}))$$

where the dependence on $\mathcal{X}$ is implicit in the computation of $\mathcal{H}(\mathbf{CL})$ and $\mathcal{S}(\mathbf{CL})$. Denoting with w_k the cardinality for a cluster $\mathbf{C}_k$, homogeneity is defined as:

$$\mathcal{H}(\mathbf{C}_k) \equiv -\frac{\sum_{\mathbf{X}\in\mathbf{C}_k}[d(\mathbf{X}, \mathbf{P}_k)]}{w_k}$$

Hence we can define clustering homogeneity as weighted average of homogeneity of clusters, and separability for a clustering as the weighted average of distances among clusters. Formally, we have:

$$\mathcal{H}(\mathbf{CL}) \equiv \frac{\sum_{i=1}^{m} w_i \cdot \mathcal{H}(\mathbf{C}_i)}{\sum_{i=1}^{m} w_i}; \;\; \mathcal{S}(\mathbf{CL}) \equiv \frac{\sum_{i=1}^{m-1}\sum_{j=i+1}^{m} w_i \cdot w_j \cdot d(\mathbf{C}_i, \mathbf{C}_j)}{\sum_{i=1}^{m-1}\sum_{j=i+1}^{m} w_i \cdot w_j} \quad (2)$$

Distance between clusters is computed as the distance between respective centroids by using (1).

3 Our Genetic Programming System for Data Clustering

Clustering Encoding. In our system, a cluster prototype is a logical formula E constituted by a variable number of predicates and a clustering is represented by a variable number of such formulas. Any GP individual encodes a clustering as a tree structure. A formula generator, based on the context–free grammar in Table 1, provides the starting population. This grammar ensures the syntactic correctness of the formulas. The tree nodes can be either terminal or nonterminal symbols. These latter are indicated in capital letter. Moreover, the terminal symbol \$ is introduced as formula delimiter. Since the grammar is non–deterministic, the action carried out by a rule is chosen based on the fixed values of probabilities shown in the table so as to reduce the probability of generating too long formulas. However, in addition an upper limit has been imposed on the total number of predicates contained in the set of formulas representing a clustering and individuals which overcome it will be generated anew. The generator starts by applying the starting rule S and then the remaining ones as long as they are called upon. The aim is to create a set of logical formulas different

in terms of size, shape and functionality. It can be noted that S allows to generate individuals composed by at least two formulas. For the predicates of each formula E_k conditions on some of the attributes a_i of the database patterns are set. When all the predicates of this formula are satisfied by the attribute values of the pattern, the formula is said to match the pattern itself. The matching between a generic pattern and the formulas is effected by an interpreter. All the formulas of the individual will be used to perform the clustering process which takes place in two steps:

1. assign all the patterns which are matched by the formulas (the formulas which match no pattern are not considered in the step 2 and they simply represent introns in the genotype):
   - a pattern is matched by one formula only: it is assigned to the related cluster;
   - a pattern is matched by more than one formula with different number of predicates: the pattern is assigned to the formula with the greatest number of matched predicates[4];
   - a pattern is satisfied by more than one formula with the same number of predicates: the pattern is classified as undefined and it is not assigned at this stage;
2. assign the patterns which are not matched by any formula of the clustering and the undefined ones:
   - for each formula, compute the average points of all the patterns assigned so far to it: this point is called *matching centroid*;
   - assign any pattern to the cluster whose matching centroid is the closest.

This process, hereinafter referred to as *clustering*, permits to combine the advantages of a clustering whose representation is based on logical formulas with those of a classical representation based on centroids. In fact, the former provides a feature extraction and correlation capability and the latter makes the process able to perform a complete clustering in which all the patterns are assigned.

Hence, based on what expressed above, the phenotype $\mathbf{F}$ is constituted by a variable number of formulas, each of which is representative of a cluster. Formally, the clustering represented by $\mathbf{F}$ is as below:

$$\mathbf{F} \equiv \{E_1, E_2, \ldots, E_{\mathcal{N}}\}$$

where $\mathcal{N}$ is the number of formulas.

Fitness Function. After performing the *clustering* process, an appropriate fitness function must be chosen to evaluate the quality of the segmentation based on $\mathbf{F}$. A penalty term linked to the number of clusters could be added in the fitness, with the aim to avoid a probable overfitting of the solution with

[4] The aim is to assign the pattern to the formula which delimits a smaller part of feature space, so to obtain a more specialized cluster.

Table 1. The grammar for the random program generator.

Rule number	Rule	Probability
1	$S \longrightarrow (Q)BA\$(Q)BAE$	1.0
2	$Q \longrightarrow L \mid M$	equiprobable
3	$L \longrightarrow C \wedge L \mid C$	0.7, 0.3
4	$M \longrightarrow C \vee M \mid C$	0.7, 0.3
5	$B \longrightarrow \vee \mid \wedge$	equiprobable
6	$A \longrightarrow (Q)BA \mid (Q)$	0.3, 0.7
7	$E \longrightarrow \$(Q)BAE \mid \$$	0.6, 0.4
8	$C \longrightarrow (PNZ)$	1.0
9	$P \longrightarrow a_0 \mid a_1 \mid \ldots \mid a_{32}$	equiprobable
10	$Z \longrightarrow 0 \mid 1 \mid 2 \mid 3$	equiprobable
11	$N \longrightarrow \geq \mid \leq \mid = \mid > \mid <$	equiprobable

respect to the given database due to uncontrolled increase in the number of clusters. Nevertheless, this would mean to establish a limit to this number, by using some a priori knowledge on the application domain, which would make our algorithm dependent on user skill. To surmount this problem, following [16], we consider as fitness function a linear combination of intra–cluster homogeneity and inter–cluster separability:

$$f(\mathbf{F}) = \mathcal{H}(\mathbf{F}) + \mu \cdot \mathcal{S}(\mathbf{F}) \tag{3}$$

where μ is a scaling factor. In this way, data clustering becomes a problem of direct maximization for homogeneity and separability independently of number of clusters $\mathcal{N}$, and depending on a scale factor liable for balance between them. Thus the fitness $f(\mathbf{F})$ does not explicitly depend on $\mathcal{N}$, yet as the parameter μ varies a control on the number of clusters will be indirectly achieved. In fact the relative weights for $\mathcal{H}$ and $\mathcal{S}$ change as μ varies. Namely it is to note that, for low values of μ, $\mathcal{H}$ is dominant in (3) and as $\mathcal{H}$ increases $\mathcal{N}$ tends to grow, while, when μ grows, the value of $\mathcal{S}$ assumes a more significant weight in (3) and, consequently, its increment tends to let $\mathcal{N}$ decrease.

To calculate more accurately $\mathcal{H}(\mathbf{F})$ and $\mathcal{S}(\mathbf{F})$, the centroids of all the actual clusters achieved at the end of the *clustering* process are evaluated and their values are used in (2).

Genetic Operators. The phenotypes are encoded as derivation trees, generated by the grammar, representing the genotypes the genetic operators work with. This encoding is very appealing in that the actions performed by the genetic operators can be easily implemented as simple operations on the trees. The new elements in the population are generated by means of two operators, *crossover* and *mutation*, which preserve the syntactic correctness of the formulas.

The crossover operates by randomly selecting both a nonterminal node in the first individual to be crossed and the same nonterminal node in the

second individual. Then, it swaps the derivation subtrees. If a corresponding nonterminal node cannot be found in the second parent, the crossover takes place on different nodes.

Differently from classical GP, the mutation works on any obtained offspring by randomly choosing a nonterminal node in the individual to be mutated, and then the corresponding production rule is activated in order to generate a new subtree. Depending on the nonterminal symbol chosen, this operation can result either in the substitution of the related subtree (macro–mutation) or in a simple substitution of a leaf node (micro–mutation).

Selection. Tournament method is taken into account in order to control loss of diversity and selection intensity.

4 Evaluation Indices and Database

4.1 Evaluation Indices

Evaluation indices can be defined and exploited to quantitatively estimate a posteriori the degree of usefulness and meaningfulness of found clusters. In general, given a clustering and a data set where classes are known, a table between the p classes and the s found clusters can be conceived. This table corresponds to a matrix $\mathbf{B}$ where element b_{ij} represents the number of patterns in the data set belonging to class i assigned to cluster j. Two indicators can be defined: the Class Addressing Index $\mathcal{I}$ and the Class Gathering Index $\mathcal{G}$. For the j-th cluster:

$$\mathcal{I}_j = \sum_{i=1}^{p} \frac{b_{ij}^2}{(\sum_{i=1}^{p} b_{ij})^2}$$

This index represents a normalized weighted average and it measures the ability of a cluster to address towards a single class. Its value is 1 in the best case, in which the cluster represents only one class, and it decreases as the number of the addressed classes increases. In the worst case its value is equal to $\frac{1}{p}$. We define $\mathcal{I}$ for the whole clustering as the weighted average for indices of each cluster:

$$\mathcal{I} = \frac{\sum_{j=1}^{s} w_j \cdot \mathcal{I}_j}{\sum_{j=1}^{s} w_j}$$

where w_j denotes the weight of j–th cluster, i.e. the number of patterns assigned to j–th cluster. Its variation range is the same of $\mathcal{I}_j$.

In a similar way, for the i–th class $\mathcal{G}$ is:

$$\mathcal{G}_i = \sum_{j=1}^{s} \frac{b_{ij}^2}{(\sum_{j=1}^{s} b_{ij})^2}$$

It indicates as a single class is subdivided among the clusters. Also in this case, its value is 1 when all the elements of one class are grouped in one cluster, while

it decreases down to $\frac{1}{s}$ as the number of clusters in which a class is subdivided increases. For all the clustering

$$\mathcal{G} = \frac{\sum_{i=1}^{p} w_i \cdot \mathcal{G}_i}{\sum_{i=1}^{p} w_i}$$

where w_i denotes the weight of i–th class, i.e. the number of patterns belonging to i–th class. The variation range is the same as that of $\mathcal{G}_i$.

For a global evaluation we take into account the so-called Correspondence Index $\mathcal{I}_c = \mathcal{I} \cdot \mathcal{G}$. Its range is $(1/p) \cdot (1/s) \leq \mathcal{I}_c \leq 1.0$.

4.2 The Database

A database, constituted by 366 clinical dermatological cases, available at UCI site [17] has been considered. Each instance has 34 attributes, of which 12 clinical and 22 hysto–pathological. Any attribute, apart from age and family anamnesis, is in the range $[0, 3]$ where 0 indicates the absence of the attribute, 3 indicates the presence at its maximum degree, while 1 and 2 denote intermediate values. Family anamnesis is a boolean attribute with value 1 if any dermatological pathology has been observed in the family, 0 otherwise. Such a database is already subdivided into 6 classes on the base of medical belief.

5 Experimental Findings

Since $d(\mathbf{X}, \mathbf{Y}) \leq 1$, for the homogeneity and the separability of a clustering, represented by a phenotype $\mathbf{F}$, we have that $-1 \leq \mathcal{H}(\mathbf{F}) \leq 0$ and $0 \leq \mathcal{S}(\mathbf{F}) \leq 1$. Therefore for the fitness it results that $-1 \leq f(\mathbf{F}) \leq \mu$. Preliminary trials have allowed to set the basic evolutionary parameters. In particular population size p_s=100, tournament size $\mathcal{T} = 10$, mutation probability p_m=0.9, crossover probability $p_c = 0.9$ and number of generations $n_g = 250$. Besides the maximum number of predicates in the set of formulas has been set equal to 200.

The experiments aim at studying the variation of $\mathcal{H}$ and $\mathcal{S}$ as a function of the scale factor μ in the range $[0.0, 1.0]$ with step 0.1. In Table 2 the average values of $\mathcal{H}$, $\mathcal{S}$ and $\mathcal{N}$ for each value of μ are reported. These values have been obtained performing 10 runs for each value of the scale factor. The objective is to find the best μ in terms of $\mathcal{H}$ and $\mathcal{S}$. In order to individuate it we have defined a theoretical 'optimal point' (shown in bold in table) obtained considering the best of homogeneity and separability among those found during our trials. Among all the different values of $\mathcal{H}$ and $\mathcal{S}$ in table, the closest to this 'optimal point', in terms of distance, has resulted to be that with $\mu = 0.7$. Table 2 shows also that as μ increases, the average number of clusters decreases which reveals that an implicit control on $\mathcal{N}$ can be actually achieved by suitably setting values for the scaling factor.

By using $\mu = 0.7$, the behavior during the best run in terms of fitness is shown in Fig. 1. In particular, on the left the average fitness of the population and the

Table 2. Average values of homogeneity and separability as a function of μ

μ	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
$\mathcal{H}$	-0,106	**-0,105**	-0,111	-0,115	-0,119	-0,120	-0,119	-0,125	-0,136	-0,140	-0,133
$\mathcal{S}$	0,234	0,237	0,244	0,248	0,253	0,254	0,252	0,259	0,265	**0,274**	0,265
$\mathcal{N}$	15.6	14.6	12.2	9.8	7.5	6.7	4.9	4.5	4.3	4.2	3.4

fitness of the best individual are reported as a function of the generation number. As it can be noted, there is a startup phase for the best fitness in which its value is almost constant and much higher than the average one. Such a behavior is due to the predominance of the separability in the fitness function which keeps low the related best number of clusters as it can be seen in Fig. 1 (right). Starting from about 40 generations this fitness increases constantly until generation 100. This is owed to the improvements of the homogeneity values as confirmed by the increase in the corresponding number of clusters with respect to the initial phase. Then the process continues with a lower increase until about 150 generations. During this phase the number of clusters remains constant. The last phase, up to generation 250, shows no significant variation in terms of the best fitness and the decrease and the stabilization in the associated number of clusters.

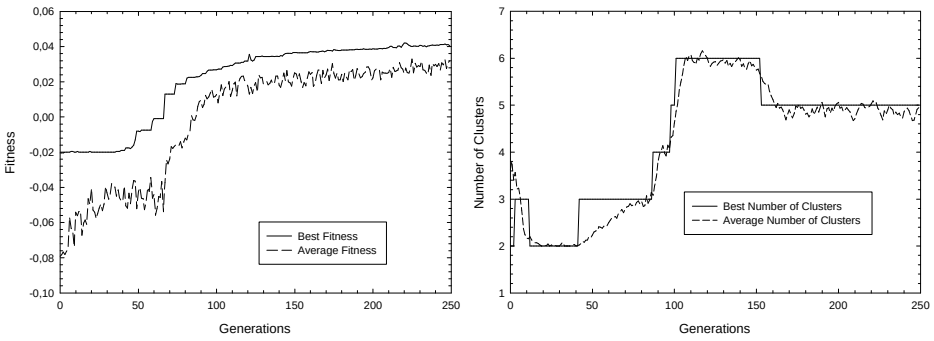

Fig. 1. Behavior of fitness (left) and number of clusters (right) during the best run.

In Table 3 the clustering obtained during the best run above is reported. For the medical database considered, it should be observed that the second and the fourth cluster address perfectly the first and the third class respectively. Almost the same holds for the first and the third cluster. Moreover it is clear that the second and the fourth classes have not been distinguished. Nonetheless, this result was expected since it is known that these classes are hard to separate by medical experts as well. This fact yields that the value of $\mathcal{I} = 0.8367$ is satisfactory but not so close to the maximum. Furthermore, it is worth observing that each class has been almost optimally gathered in one cluster. This is confirmed by the very good value of $\mathcal{G} = 0.989$. Thus the global index $\mathcal{I}_c$ is

Table 3. Classes–clusters ($p\backslash s$) relationship for the solution with $\mu = 0.7$

$p\backslash s$	1	2	3	4	5
1	0	112	0	0	0
2	0	0	0	0	61
3	0	0	0	72	0
4	0	0	0	0	49
5	50	0	0	0	2
6	0	0	19	0	1

Table 4. Average index values as a function of μ

μ	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
$\mathcal{I}$	0.5945	0.925	0.803	0.927	0.746	0.8154	0.8274	0.8167	0.665	0.6642	0.615
$\mathcal{G}$	0.835	0.735	0.783	0.816	0.704	0.938	0.952	0.979	0.949	0.916	0.933
$\mathcal{I}_c$	0.4964	0.6799	0.6287	0.7564	0.5252	0.7648	0.7877	0.7995	0.6311	0.6084	0.5738

equal to 0.8275. These values assure that the clustering achieved is effective.

As an example of the clustering, we report in the following only the formulas representative of the first two clusters in Table 3:
Cluster 1: $((a_{17} = 2) \vee (a_{10} = 1) \vee (a_{15} > 0)) \wedge ((a_{27} = 1) \vee (a_6 > 1)) \wedge$
$((a_{16} = 2) \vee (a_2 = 2) \vee (a_{30} = 3) \vee (a_{19} = 1) \vee (a_3 = 2) \vee (a_{15} < 2)) \wedge (a_{14} \geq 2)$
Cluster 2: $(a_{21} > 1) \wedge ((a_{22} > 1) \vee (a_{26} > 1) \wedge (a_0 > 1) \vee (a_{28} > 0) \vee (a_{21} < 3) \vee$
$(a_0 < 3) \vee (a_{12} = 1) \vee (a_{18} > 1) \vee (a_{31} \leq 2) \vee (a_0 < 2) \vee (a_{11} > 1) \vee (a_{24} > 1)) \wedge$
$((a_{29} = 0) \vee (a_{18} < 2) \vee (a_{10} < 3) \vee (a_{12} < 2))$

As it is evident this novel representation has the advantage of providing explicit information both on the features of the patterns satisfying each formula and on their values.

Let us use $\mathcal{I}_c$ index to evaluate the found solutions from a clustering standpoint. It permits to estimate the utility of detected clusters in addressing towards the classes. In Table 4 the average values of all the indices, related to the best solutions in terms of fitness during 10 runs, are reported as a function of μ. It should be observed that the value $\mathcal{I}_c$ obtained for $\mu = 0.7$ is always the best. This seems to validate a posteriori its choice.

6 Conclusions and Future Work

In this paper, the flexibility of a Genetic Programming framework in performing automatic clustering has been exploited. The implemented system is capable of detecting clusters orienting towards classes. As a data clustering algorithm, satisfactory results have been achieved. Furthermore, thanks to GP, an innovative mode of representing clusters, based on logical formulas, has been introduced. This has not only allowed to perform an original clustering process which exploits the advantages of both a new representation and a classical one,

but it has also permitted the discovery of common relationships of patterns belonging to a same cluster.

The system will be tested on different databases to further ascertain its effectiveness. Besides, it could be parallelized with the scope of achieving an improvement both in performance and in solution quality.

References

1. Fayyad, U. M., Piatetsky-Shapiro, G., Smith, P.: From data mining to knowledge discovery: an overview. In Advances in Knowledge Discovery and Data Mining. U. M. Fayyad et al. (eds) AAAI/MIT Press (1996) 1–34
2. Hand, D. J., Mannila, H., Smyth, P.: Principles of data mining. MIT Press (2001)
3. Han, J., Kamber, M.: Data mining: concepts and techniques. Morgan Kaufmann (2001)
4. Zhang, T., Ramakrishnan, R., Livny, M.: BIRCH: an efficient data clustering method for very large databases. In Proc. ACM SIGMOD Int. Conf. on Management of Data (1996) 103–114
5. Guha, S., Rastogi, R., Shim, K.: CURE: an efficient clustering algorithm for large databases. In Proc. ACM SIGMOD Int. Conf. on Management of Data (1998) 73–84
6. Aggarwal, C., Yu, P. S.: Finding generalized projected clusters in high dimensional spaces. In Proc. ACM SIGMOD Int. Conf. on Management of Data (2000) 70–81
7. Bock, H. H.: Probability models in partitional cluster analysis. In Developments in Data Analysis. A. Ferligoj, A. Kramberger (eds) (1996) 3–25
8. Fraley, C., Raftery, A.: How many clusters? Which clustering method? Answers via model–based cluster analysis. The Computer Journal **41** (8) (1998) 578–588
9. Lee, C. Y., Antonsson, E. K.: Dynamic partitional clustering using evolutionary strategies. In Proc. of the 3rd Asia–Pacific Conference on Simulated Evolution and Learning. IEEE Press, Nagoya, Japan (2000)
10. Jain, A. K., Murty, M. N., Flynn, P. J.: Data clustering: a review. ACM Computing Surveys **31** (3) (1999) 264–323
11. Hall, L. O., Ozyurt, B., Bezdek, J. C.: Clustering with a genetically optimized approach. IEEE Trans. on Evolutionary Computation **3** (2) (1999) 103–112
12. Sarafis, I., Zalzala, A. M. S., Trinder, P. W.: A genetic rule–based data clustering toolkit. In Proc. of the IEEE Congress on Evolutionary Computation (2002) 1238–1243
13. Cristofor, D., Simovici, D. A.: An information–theoretical approach to clustering categorical databases using genetic algorithms. In Proc. of the Workshop on Clustering High Dimensional Data and its Applications. In Proc of the Second SIAM International Conference on Data Mining. Washington (2002) 37–46
14. Babu, G. P., Marty, M. N.: Clustering with evolutionary strategies. Pattern Recognition **27** (2) (1994) 321–329
15. Koza, J. R.: Genetic Programming: on programming computers by means of natural selection and genetics. The MIT Press, Cambridge, MA (1992)
16. Yip, A. M.: A scale dependent data clustering model by direct maximization of homogeneity and separation. Proc. Mathematical challenges in scientific data mining IPAM. 14-18 January (2002)
17. Murphy, P. M., Aha, D. W.: UCI Repository of machine learning databases. University of California, Department of Information and Computer Science. www.ics.uci.edu/∼mlearn/MLRepository.html