Alternative Clustering by Utilizing Multi-Objective Genetic Algorithm with Linked-List Based Chromosome Encoding

Jun Du[1], Emin Erkan Korkmaz[3], Reda Alhajj[1,2], and Ken Barker[1]

[1] Department of Computer Science, University of Calgary
Calgary, Alberta, Canada
{jundu, alhajj, barker}@cpsc.ucalgary.ca

[2] Department of Computer Science, Global University
Beirut, Lebanon

[3] Department of Computer Engineering, Yeditepe University,
Kadikoy, Istanbul, Turkey
ekorkmaz@cse.yeditepe.edu.tr

Abstract. In this paper, we present a linked-list based encoding scheme for multiple objectives based genetic algorithm (GA) to identify clusters in a partition. Our approach obtains the optimal partitions for all the possible numbers of clusters in the *Pareto Optimal* set returned by a single genetic GA run. The performance of the proposed approach has been tested using two well-known data sets, namely *Iris* and *Ruspini*. The obtained results are promising and demonstrate the applicability and effectiveness of the proposed approach.

keywords: clustering, genetic algorithms, linkage-encoding, k-means, multi-objective optimization.

1 Introduction

In this paper, a new scheme is proposed for encoding clustering solutions into chromosomes. The proposed representation forms a linked-list structure for objects in the same cluster. The genetic operators modify the chromosomes by altering the links. Also, we deal with the partitional clustering problem by using a multi-objective GA [15] to minimize *Total Within Cluster Variation (TWCV)*, together with the number of clusters. TWCV [17] is a measure which denotes the sum of the average distance of cluster elements to cluster center. If this measure is used as the sole objective in the search, GA will tend to reduce the size of the clusters and eventually will form clusters with single elements where the variation turns out to be zero. Hence, traditionally, a prior specified number of clusters is needed for GA based k-clustering approaches treating TWCV as the single objective function. The other objective (minimizing the number of clusters) effectively handles this.

The new representation proposed in this paper is able to encode the solution space in fixed-length chromosomes. It enables an efficient exploration of the solution space. Together with the use of multi-objective GA, this approach can be extended to deal with the general clustering problem where the optimum number of clusters is unknown. The *Pareto Optimal* set [15], obtained at the end of the run provides solutions with the optimum TWCV for various potential numbers of clusters.

Two well-known data sets *Iris* [3] and *Ruspini* [22] have been used in the experiments to demonstrate the applicability, usefulness and effectiveness of the proposed approach. These data sets have been widely used as benchmark problems for testing different techniques, and their corresponding optimal clustering is known. Hence, it is easy to evaluate the performance of a clustering method by using these data sets.

The rest of the paper is organized as follows. The objective functions used are discussed in Section 2. A closer look at the proposed approach is presented in Section 3. The experimental results obtained on two well-known data sets are reported in Section 4. Section 5 is conclusions.

2 The Objective Functions

Many optimization problems are multi-objective by nature. The classical approach to such problems is to use a single objective function which is obtained by a linear combination (weighted sum) of multiple objectives. Another approach is to treat different objectives as different constraints and use thresholds and penalties during the search. However, the usage of weights and penalties has been clearly proved problematic in the domain of GA. In our approach, the *Niched Pareto Genetic Algorithm* described in [15] is used in order to minimize the following two objectives.

1. Total within cluster variation (TWCV), which has been effectively used for the k-clustering problem.
2. Number of clusters.

The formal definition of TWCV is given in [17] as follows. Let the clustering problem be partitioning n objects, each has d different properties, into k different groups. So, each object can be represented as a vector with dimension d, and the collection of these objects would be a matrix X, where entry x_{ij} denotes the j^{th} property of the i^{th} object. Then, another matrix W can be defined as:

$$w_{ik} = \begin{cases} 1, if\ i^{th}\ pattern\ belongs\ to\ k^{th}\ cluster. \\ 0, otherwise \end{cases} \tag{1}$$

The following two properties will hold for the new matrix; $w_{ij} \in \{0,1\}$ and $\sum_{k=1}^{K} w_{ij} = 1$, where K is the total number of clusters.

Let $c_k = \{c_{k1}, c_{k2}..., c_{kd}\}$ denote the center of the k^{th} cluster, then

$$c_{kj} = \frac{\sum_{i=1}^{n} w_{ik} x_{ij}}{\sum_{i=1}^{n} w_{ik}}. \tag{2}$$

The within-cluster variation (WCV) of the k^{th} cluster can be defined as

$$S^{(k)}(W) = \sum_{i=1}^{n} w_{ik} \sum_{j=1}^{d} (x_{ij} - c_{kj})^2 \tag{3}$$

Lastly, TWCV is defined as

$$S(W) = \sum_{k=1}^{K} S^{(k)} = \sum_{k=1}^{K} \sum_{i=1}^{n} w_{ik} \sum_{j=1}^{d} (x_{ij} - c_{kj})^2 \tag{4}$$

3 A Closer Look at the Proposed Approach

The most straightforward and the most widely used encoding scheme is *Group Number Encoding* [16]. In this scheme, the value of each gene represents the membership of an object to one of the clusters. Let the set of objects to be clustered in k groups be $\mathcal{O} = \{o_1, o_2, ..., o_n\}$. Since one gene is reserved for each object, the length of the chromosomes will be n. Let V be a function denoting the value of a gene in a chromosome. If $\mathcal{C} = \{g_1, g_2, ..., g_n\}$ is a chromosome in the population, where $\forall g_i \in \mathcal{C}, 1 \leq V(g_i) \leq k$, then $V(g_i)$ will denote the cluster number for object o_i. Two objects, o_i and o_j will be in the same cluster if and only if $V(g_i) = V(g_j)$.

For example, the sample chromosome 2316736211 would encode the clustering solution where the first object is in cluster 2, the second in 3 and so on. However, it is possible to have multiple distinct chromosomes for the same solution with this encoding. In a clustering process, the naming or the ordering of the clusters is irrelevant. For instance, renaming cluster 2 to cluster 5 in chromosome 2316736211 creates a new chromosome 5316736511. However, both chromosomes are mapping to the same clustering solution. The drawbacks of this traditional encoding are presented in [8], and it is pointed out in [20] that this encoding is against the minimal redundancy principles set for encoding scheme design. The remedy proposed in [8] is to use a length variable encoding scheme. It reduces redundancy of chromosome population, but in the meantime it adds redundancy inside a chromosome; it needs more genes to encode a solution than traditional encoding. The other deficiency of the length variable encoding is that it cannot take advantage of conventional simple crossover and mutation operators. This gives advantage to the *Linear Linkage Encoding* presented in this section; it is a fixed length encoding scheme without any type of redundancy.

Under linkage encoding scheme, although each gene still stores an integer, the value of the gene no longer directly denotes the membership of an object but its fellowship - this is the fundamental difference between the group number encoding and the linkage encoding. Each gene is a link from an object to another object of the same cluster. Given n objects, any partition on them can be described as a chromosome of length n. Two objects are in the same group if either object can be directed to the other object via the links. Without any constraint, the state of redundancy is just as bad as that of the group number encoding because the number of feasible chromosomes is still n^n.

Linear linkage encoding is a restricted linkage encoding. Let the n genes be indexed inside a chromosome from 1 to n. The value of each gene in LL chromosome denotes the index of a fellow gene where the objects that corresponding to these two genes would be in the same cluster. We can also treat the stored index as an out-link from a node, and if a gene stores its own index, it depicts an ending node. To qualify an unrestricted linkage chromosome as a valid linear linkage encoding chromosome, there are two constraints the chromosome must comply to:

1. The integer value stored in each gene is greater than or equal to its index but less than or equal to n.
2. No two genes in the chromosome have the same value with the exception that at most two genes can have the same integer value if the integer is the index of an ending node.

Formally, let the set of objects to be clustered be $\mathcal{O} = \{o_1, o_2, ..., o_n\}$ and let $\mathcal{C} = \{g_1, g_2, ..., g_n\}$ be a sample chromosome in the population. Assume V is a function that denotes the value of a gene and I is the function which returns its index. Then, the following two properties hold for the LL encoding.

$$\forall g_i \in \mathcal{C} \left[I(g_i) \leq V(g_i) \leq n \right]. \tag{5}$$

$$\begin{aligned} \forall g_i, g_j \in \mathcal{C}[V(g_i) = V(g_j) \implies (i = j) \\ \vee((i > j)) \wedge (V(g_i) = I(g_i)) \\ \vee((i < j) \wedge (V(g_j) = I(g_j))]. \end{aligned} \tag{6}$$

The boolean function ($\varphi : \mathcal{O X O} \mapsto \{True, False\}$), which would determine if two given objects are in the same cluster or not, can be recursively defined. If o_i and o_j are two objects where $i < j$, then

$$\varphi(o_i, o_j) = \begin{cases} [V(g_i) = I(g_j)] \vee \\ \exists g_k [(i < k < j) \wedge \varphi(o_i, o_k) \wedge V(g_k) = I(g_j)] \end{cases} \tag{7}$$

Linear linkage encoding gets its name because objects in a cluster construct a pseudo linear path with the only loop allowed being a self loop link to mark the last node. It can be represented by the *labeled oriented pseudo* (LOP) graph.

A LOP graph is a labeled directed graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where $\mathcal{V}(\mathcal{G}) = \{v_1, v_2, ..., v_n\}$. A composition of $\mathcal{G}$ is a partition of $\mathcal{V}(\mathcal{G})$ into disjointed oriented pseudo path graphs $\mathcal{G}_1, \mathcal{G}_2, ..., \mathcal{G}_m$ with the following properties:

1. Disjoint paths: $\bigcup_{i=1}^{m} \mathcal{V}(\mathcal{G}_i) = \mathcal{V}(\mathcal{G})$ and for $i \neq j, \mathcal{V}(\mathcal{G}_i) \bigcap \mathcal{V}(\mathcal{G}_j) = \emptyset$
2. Non-backward oriented edges: If there is an edge e directed from vertex v_l to v_k, then $l \leq k$.
3. Balanced connectivity:
   (a) $|\mathcal{E}(\mathcal{G})| = |\mathcal{V}(\mathcal{G})|$
   (b) Each $\mathcal{G}_i$ must have only one ending node with a self referencing directed edge exists. The ending node has an indegree of 2 and an outdegree of 1.

(c) Each $\mathcal{G}_i$ must have only one starting node whose indegree is 0 and outdegree is 1.

(d) All other $|\mathcal{V}(\mathcal{G}_i)|-2$ vertexes in $\mathcal{G}_i$ have both their indegree and outdegree equal to 1.

Theorem 1: Given a set of objects $\mathcal{S}$, there is one to one mapping between the chromosomes of LL encoding and the possible partition schemes.

In order to prove this theorem the following lemmas are used.

Lemma 1: Linear linkage encoding is an implementation of the LOP graph.

Lemma 2: Given a set of objects $\mathcal{S}$, there exists one and only one composition of LOP graph $\mathcal{G}(\mathcal{V},\mathcal{E})$ for each partition scheme of $\mathcal{S}$, where $|\mathcal{V}| = |\mathcal{S}|$.

Note that, there is only one possible ascending order within clusters for a possible partition scheme. Thus, there exists only one composition of G for each partition. Reversely, a LOP graph represents only a single partition scheme by definition. Based on Lemmas 1 and 2, it can be claimed that LL encoding makes a one-to-one mapping between the chromosomes and clustering solutions.

Corollary : The number of chromosomes corresponding to all possible partition schemes is given by the n^{th} *Bell number*.

The number of ways a set of n elements can be partitioned into non-empty subsets is called a Bell number [4]. According to Theorem 1, there is one-to-one correspondence between the chromosomes of LL encoding and the possible partition schemes. Hence, the number of chromosomes in consideration would be denoted by the n^{th} Bell number $B(n)$, too. Compared to LL encoding scheme, traditional group number encoding demands GA to work in a solution space of $\frac{n^n}{B(n)}$ times larger. When n is 10, $\frac{n^n}{B(n)}$ is about 10^5.

Although LL encoding keeps only fellowship in genes, it also implies the membership of each object. Since each cluster must have one starting node and one ending node, both nodes can be used to identify a cluster. In practice, ending node is treated as the membership identifier for clusters because it is easier to detect. Apparently, finding the membership of an object in LL encoding requires only linear time.

The initial population should include diverse chromosomes. It is intuitive to achieve this goal by generating random chromosomes, which means each gene in a chromosome is assigned an integer randomly selected from the range 1 to n, where n is the number of objects to be clustered. However, the chromosomes generated this way may violate the restrictions of linear linkage encoding. Based on the first LL encoding constraint, each integer should be between its index and the maximum integer index number, inclusive. Therefore, a chromosome generator for creating each gene based on this constraint would be a better choice for diversity. Note that, the chromosomes produced this way still would not be fully complied with the constraints laid for linear linkage encoding. Obviously, backward links are prevented with this generator, but multiple nodes can link to the same node, violating the second constraint.

Note that multiple links are allowed during the initialization process. Later, we will see backward links in a chromosome emerge in the process of the mutation operation. Therefore, a recovery process is needed after the constructors, and later other GA operators are employed to rectify a chromosome into its legitimate format. The Rectifying algorithm used for the recovery process involves two correction steps. First, backward links are eliminated from a chromosome. Then, multiple links to a node (except for the ending nodes) are replaced with one link in and one link out.

The selection process is very similar to that of Niched Pareto GA described in [15]. A chromosome is said to be fitter or to dominate another one when it is superior to the latter in terms of all the objective functions used. If only a part of the objective values of one chromosome are better than the other's, neither chromosome is deemed dominant to the other. A chromosome can be compared to a set of chromosomes. It is dominated by the set if any individual in the set is fitter than it. Otherwise, the chromosome is not dominated by the set.

When two randomly selected chromosomes competing for a spot in the parent pool, they are not directly compared with each other. Rather, each is compared to a comparison set of chromosomes sampled from the current generation. If one of the competing chromosomes, say A, is dominated by the comparison set and the other chromosome, say B, is not dominated, then B advances to the parent pool. However, when both A and B are either dominated or not dominated by the set, the niche count of each chromosome is compared. The chromosome with the smaller niche count gets advantage. Niche count is an indicator of the solution density around a chromosome in a certain solution population. This approach encourages even distribution of solutions in the GA population [15].

In each generation, the Pareto dominant set is achieved through a search in the whole population. Every individual is compared with the rest. If a chromosome is not dominated by the rest, it is copied to the Pareto dominant set. The Pareto dominant set of the last generation contains the optimal solution.

In our experiments, one point crossover is adapted. The operation both allows different clusters to exchange partial contents and may split a cluster into two.

The classical mutation was implemented for LL-encoding and the test results were not encouraging. With the classical mutation, the out-link of a node is very likely to change to a different one, but the new out-link might still point to the same cluster. This results in no change in the chromosome after being rectified, and hence lessens the affect of the mutation during the search. The solution is to make sure that the mutated gene gains a link to a different cluster instead of just a different node. Also, when the mutated gene is again assigned to its original cluster, that cluster is split into two. Hence, the mutation might have two different effects on the chromosome; a sub-group of objects can be moved to a new cluster or a cluster can be split into two. This new mutation method changes the membership of set of objects rather than just a single object.

4 Experimental Results

In this section, we report the result of our experiments on two widely used data sets, namely Iris and Ruspini. The former is a real dataset recording 150 observations of the three species (Setosa, Versicolor, and Virginica) of iris flowers, all

Table 1. Genetic parameters used for the experiments.

Parameter	**Value**
Number of Experiments	10
Number of Generations	2000
population size (Iris)	800
population size (Ruspini)	500
Niche Comparison Size (selection)	5
Nitch Radius	5
Nitch Count Size	25
Crossover Rate	0.9
Mutation Rate	0.2

described in four features. The latter is an artificial dataset of two attributes. Its 75 data points naturally form 4 clusters. Originally data points in both datasets are sorted so that the clusters are easily perceived. To obtain unbiased result, we reorganized the datasets and all the data points are randomly placed. In addition, data standardization process is applied to the dataset to neutralize the effects brought by data attributes of disparage magnitude. We divide each data element by the maximum value of its dimension to scale all data elements ranging from 0 to 1. In our experiments, two GAs are developed. Apart from the encoding schemes, all GA operators are kept the same as described in Section 3. The genetic parameters are fixed for all GA runs and they are presented in Table 1.

Note that the multi-objective GA tries to minimize TWCV for all possible number of clusters. For the Iris data set, the possible number of clusters ranges from 1 to 150. The single cluster is the case where all instances are placed into the same cluster, and each instance is considered as a separate cluster when number of clusters increases to 150. This range is between 1 and 75 for the Ruspini dataset since the number of instances is 75 in this domain. Note that the optimal number of clusters for Iris and Ruspini is 3 and 4, respectively. Hence, TWCV values obtained for smaller number of clusters is of more interest.

For Iris, the change in TWCV is quite stable down to 3 clusters. However, there is a considerable leap between TWCV values of 2 and 3 clusters. The same is valid for Ruspini between 3 and 4 clusters. Hence, it is possible to derive conclusions about the optimum number of clusters by considering the pareto optimal set obtained at the end of the GA search. In both data sets the optimum clusters are well separated from others. We realize that in a domain where the cluster borders are not very clear, the leap at the optimum number of clusters may not be as clear as the result obtained in these two domains.

Note that Ruspini dataset has only two attributes. It is easy to display the clusters obtained on this data set as two-dimensional diagrams. In Figure 1 the best partitions obtained by the *Linkage encoding* are presented for 3, 4 and 5 clusters. From Figure 1, it can be easily realized that the natural partition of the dataset is formed by four clusters (Figure 1-b). However, the partitions obtained

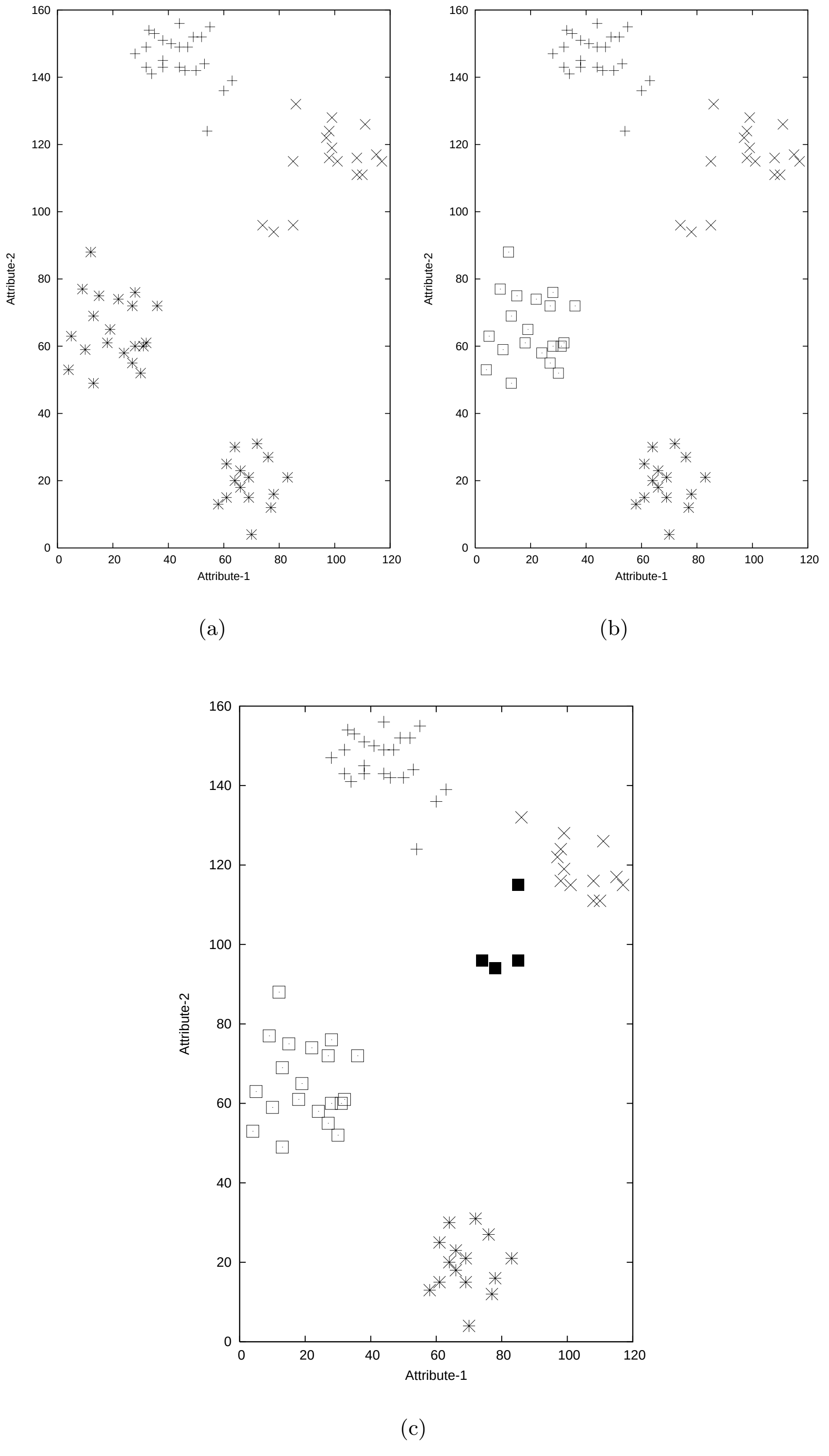

Fig. 1. The best TWCV values obtained when the number of clusters is: a) 3; b)4; c) 5

when the number of clusters is decreased to 3 or increased to 5 are also plausible. When the number of clusters is 3, two of the clusters that appear in the natural solution merge into a single cluster (Figure 1-a). On the other hand, one of the clusters in the natural solution splits into two when the number of clusters is 5. In this case, a new cluster is created with the elements that seem to be a bit more separate compared to the other elements in the original class (Figure 1-c).

5 Conclusions

In this paper, a new encoding scheme is proposed for the application of GA to the clustering problem. This new scheme has been successfully used with the multi-objective GA which is a powerful optimization technique. The results obtained on two well-known data sets provide a good insight about the importance and effectiveness of the new scheme. The analysis carried out clearly notifies that the new scheme is applicable, useful and effective. Although some extra processes are needed in order to keep the redundancy low, it has been observed that the computational cost of these processes is not significant.

The leap in TWCV after the optimum number of clusters seems to be an important issue about the proposed technique. The experiments demonstrate that it is expected to observe such a leap for datasets with well separated clusters. It would be interesting to observe the change in TWCV, in domains where cluster borders are not clear. In such domains, probably it would not be possible to directly observe the optimum number of clusters. However, an automatic analysis of the change in TWCV values might be helpful to determine the optimum point.

References

1. a Pen, J. Lozano, and J. Larran. An empirical comparison of four initialization methods for the k-means algorithm. *Pattern Recognition Letters*, 20(10):1027–1040, 1999.
2. Pavel Berkhin. Survey of clustering data mining techniques. Technical report, Accrue Software, San Jose, CA, 2002.
3. C. L. Blake and C. J. Merz. UCI repository of machine learning databases, 2000.
4. Andrew Bremner. Reviews: *The Book of Numbers*, by John Horton Conway and Richard K. Guy. *American Mathematical Monthly*, 104(9):884–??, November 1997.
5. Donald S. Burke, Kenneth A. De Jong, John J. Grefenstette, Connie Loggia Ramsey, and Annie S. Wu. Putting more genetics into genetic algorithms. *Evolutionary Computation*, 6(4):387–410, Winter 1998.
6. Rowena Marie Cole. Clustering with genetic algorithms. Master's thesis, Nedlands 6907, Australia, 1998.
7. I. Dhillon, J. Fan, and Y. Guan. Efficient clustering of very large document collections. In R. Grossman, C. Kamath, and R. Naburu, editors, *Data Mining for Scientific and Engineering Applications*. Kluwer Academic Publishers, 2001.
8. E. Falkenauer. *Genetic Algorithms and Grouping Problems*. John Wiley&Sons, 1998.
9. Emanuel Falkenauer. A new representation and operators for genetic algorithms applied to grouping problems. *Evolutionary Computation*, 2(2):123–144, 1994.

10. David Gibson, Jon M. Kleinberg, and Prabhakar Raghavan. Clustering categorical data: An approach based on dynamical systems. *VLDB Journal: Very Large Data Bases*, 8(3–4):222–236, February 2000.
11. D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.
12. Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. CURE: an efficient clustering algorithm for large databases. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data, Seattle, Washington, USA*, volume 27(2), pages 73–84, 1998.
13. J. Han and M. Kamer. *Data Mining: Concepts and Techniques.* Morgan Kaufmann, 2001.
14. John Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, 1975.
15. Jeffrey Horn, Nicholas Nafpliotis, and David E. Goldberg. A Niched Pareto Genetic Algorithm for Multiobjective Optimization. In *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, volume 1, pages 82–87, 1994.
16. Donald A. Jones and Mark A. Beltramo. Solving partitioning problems with genetic algorithms. In Lashon B. Belew, Richard K.; Booker, editor, *Proceedings of the 4th International Conference on Genetic Algorithms*, pages 442–449. Morgan Kaufmann, July 1991.
17. K. Krishna and M. Murty. Genetic k-means algorithm. *IEEE Transactions on Systems, Man, and Cybernetics - PartB: Cybernetics*, 29(3):433–439, 1999.
18. U. Maulik and S. Bandyopadhyay. Genetic algorithm-based clustering technique. *Pattern Recognition*, 33:1455–1465, 2000.
19. R. Ng and J. Han. Efficient and effective clustering method for spatial data mining. In *Proc. of 1994 Int'l Conf. on Very Large Data Bases (VLDB'94)*, pages 144–155, September 1994.
20. Nicholas J. Radcliffe. Forma analysis and random respectful recombination. In Lashon B. Belew, Richard K.; Booker, editor, *Proceedings of the 4th International Conference on Genetic Algorithms*, pages 222–229. Morgan Kaufmann, July 1991.
21. J. T. Richardson, M. R. Palmer, G. Liepina, and M. Hilliard. Some guidlines for genetic algorithms with penalty functions. In *Proc. of the 3rd Int. Conf. on Genetic Algorithms*, pages ?–? Morgan Kaufman, 1989.
22. E.H. Ruspini. Numerical methods for fuzzy clustering. *Inform. Sci.*, 2(3):19–150, 1970.
23. I. Sarafis, A. M. S. Zalzala, and P. Trinder. A genetic rule-based data clustering toolkit. In *Proceedings of the 2002 Congress on Evolutionary Computation CEC2002*, pages 1238–1243, 2002.