ELSEVIER

Pattern Recognition Letters 20 (1999) 911–918

Pattern Recognition Letters

www.elsevier.nl/locate/patrec

Applying genetic algorithms to search for the best hierarchical clustering of a dataset

J.A. Lozano [*], P. Larrañaga

Departamento de Ciencias de la Computación e Inteligencia Artificial, Universidad del País Vasco, Aptdo. 649, 20080 San Sebastian, Donostia, Spain

Received 19 February 1998; received in revised form 19 May 1999

Abstract

We apply genetic algorithms to get the optimal, in a least squares sense, hierarchical clustering of a dataset. We base this on the bijection between the set of hierarchical classifications of a dataset and the set of ultrametric distances. This bijection makes it possible to measure how good a hierarchical classification is, by calculating the L_2 norm between the ultrametric distance matrix associated with the hierarchical classification and the proximity matrix of the dataset. Our results are shown to improve on other methods which have been proposed, based on the ultrametric. © 1999 Elsevier Science B.V. All rights reserved.

Keywords: Hierarchical clustering; Genetic algorithms; Least squares

1. Introduction

Hierarchical clustering (Gordon, 1991, 1996; Jain and Dubes, 1988; Kaufman and Rousseeuw, 1990) is a technique of exploratory data analysis. It builds a hierarchically-nested sequence of partitions of a dataset commonly represented in a rooted tree diagram.

An n-tree in a set Ω is a subset $T \subset \boldsymbol{P}(\Omega)$ of parts of Ω such that:
1. $\Omega \in T$,
2. $\emptyset \notin T$,
3. $\forall\, \omega \in \Omega \Rightarrow \{\omega\} \in T$,
4. $\forall\, A, B \in T \;\Rightarrow\; A \cap B \in \{\emptyset, A, B\}$.

Formally a hierarchical classification or dendrogram of Ω is a pair (T, h) formed by an n-tree T together with a non-negative function h that assigns a number to each node of T satisfying:
1. $h(A) = 0 \Longleftrightarrow A = \{\omega\}$ for some $\omega \in \Omega$,
2. If $A \cap B \neq \emptyset$ then $A \subset B \Longleftrightarrow h(A) \leqslant h(B)$.

There are two main sorts of algorithms which build a hierarchical clustering of a dataset: the *agglomerative algorithms* and the *divisive algorithms*.

The approach that seems to have received more attention is the agglomerative algorithm (Kaufman and Rousseeuw, 1990; Gordon, 1996). In agglomerative algorithms, each point of the dataset forms a cluster at first, and at each step of the algorithms two clusters are joined to form a new cluster. Different results can be obtained using different criteria to choose the clusters that have to be joined. A summary of these kinds of criteria can

[*] Tel.: +34-43-448-000 ext. 5034; fax: +34-43-219-306; e-mail: lozano@si.ehu.es

PII: S 0 1 6 7 - 8 6 5 5 (9 9) 0 0 0 5 7 - 4

be found in (Chandon and Pinson, 1980) and most of them can be found implemented in commercial software (SPSS, SAS, BMDP, etc.).

Divisive algorithms start with one cluster containing the whole dataset. At each step of the algorithm a cluster is divided into two new clusters until there is only one point in each cluster. Again, different results can be obtained using different criteria in dividing the clusters. Some of them can be found in (Kaufman and Rousseeuw, 1990).

Both agglomerative and divisive algorithms have the same characteristic in that they are greedy algorithms, i.e., they build the *n*-tree choosing the best possibility at any stage without taking into account any global criterion related to the whole structure. In this work, we seek to avoid locally optimal solutions.

Given a hierarchical classification (T, h) of a dataset Ω, it is possible to build a distance U from it, in the dataset, as follows:

$$U(\omega, \omega') = \min_{A \in T}\{h(A) \mid \omega, \omega' \in A\}.$$

This distance has the following special properties:
1. $\forall\ \omega, \omega' \in \Omega \Rightarrow U(\omega, \omega') = U(\omega', \omega)$,
2. $\forall\ \omega \in \Omega \Rightarrow U(\omega, \omega) = 0$,
3. $\forall\ \omega, \omega', \omega'' \in \Omega$
   $\Rightarrow U(\omega, \omega'') \leqslant \max\{U(\omega, \omega'), U(\omega', \omega'')\}$.

The latter property is called the *ultrametric property* and this is the name given to this kind of distance, *ultrametric distance*. An important result given in separate papers in 1967 (Hartigan, 1967; Jardine et al., 1967; Johnson, 1967) establishes the existence of a bijection between the set of hierarchical classifications of a dataset and the set of ultrametric distances. Therefore, given a hierarchical classification, it is possible to get its associated ultrametric distance and compare it with the initial dissimilarities (hereafter, we can assume that the proximities are dissimilarities, the similarity case being slightly different) of the dataset. Therefore, the better the ultrametric distance fits to the dissimilarities of the dataset, the better the hierarchical classification will be. As a result, the problem of searching for the best hierarchical clustering can be set as an optimisation problem:

given a dissimilarity of a dataset, search for the ultrametric distance closest to it.

If we can find this closest ultrametric distance, we will then find the best hierarchical classification of the dataset.

Different criteria could be used to measure the fitting between an ultrametric distance and a set of dissimilarities, many of them can be found in (Carroll and Pruzansky, 1975), but the one that has received the most attention is the L_2 or Euclidean norm. In this case the optimisation problem can be rewritten as that of searching for the least squares ultrametric distance of a dataset. We use this norm in our approximation. Note that the search for the least squares ultrametric distance of a set of dissimilarities is equivalent to looking for the ultrametric distance with the biggest cophenetic correlation coefficient (De Soete, 1984).

Genetic algorithms (Goldberg, 1989; Holland, 1975) (GAs) are probabilistic search algorithms which simulate natural evolution. They are based on the mechanics of natural selection and genetics. They combine 'survival of the fittest' among string structures with a structured yet randomised information exchange. In GAs the search space of a problem is represented as a collection of individuals. The individuals are represented by character strings, which are referred to as *chromosomes*. The aim is to find the individual from the search space with the best 'genetic material'. The quality of an individual is measured with an objective function. The part of the search space to be examined in each iteration is called the *population*. A GA works approximately as follows. To begin with, the initial population is chosen at random, and the quality of each of its individuals is determined. Next, in every iteration, parents are selected from the population. These parents produce children, which are added to the population. For all the newly created individuals of the resulting population a near to zero mutation probability will exist, i.e., the newly created individual could change their hereditary distinctions. The population is then reduced to its initial size by removing some individuals according to a selection criterion. One iteration of the algorithm is referred to as a *generation*.

2. Previous approaches to the problem

The problem of searching for an ultrametric distance closest to the dissimilarity of a dataset was set up in an early paper (Hartigan, 1967). Hartigan tried to carry out this task in a local form. He defined a group of local operations over a tree structure, his aim being to get the best tree for these local operations. He raised the idea of searching the complete search space to find the best tree for the first time. This work was further improved in (Hartigan, 1985), but it still developed a local algorithm.

An important paper in this topic was presented by Chandon et al. (1980). They developed a branch-and-bound algorithm which could find the optimal least squares ultrametric of a dataset. The problem of this approximation is that it is very time-consuming, so it can only be applied to small datasets (size lower than 15). Their algorithm is based on an important property that will be used by us. If we denote the different element pairs' set of the dataset $S = \{(i,j)\,|\,\omega_i \neq \omega_j\}$, then any ultrametric distance $U = [\delta_{i,j}]_{i,j=1,2,\ldots,n}$ can be decomposed into a preorder relation on S denoted by Γ:

$$\Gamma_1 < \Gamma_2 < \cdots < \Gamma_k$$

and a sequence of numbers

$$\delta_1 < \delta_2 < \cdots < \delta_k,$$

such that

$$\forall\ (i,j) \in \Gamma_r \Rightarrow \delta_{i,j} = \delta_r \quad \forall r \in \{1,2,\ldots,k\}.$$

The above mentioned property says that if Γ is a solution of the problem, then

$$\delta_r = (1/n_r) \sum_{(i,j)\in\Gamma_r} d_{i,j} \quad \forall r \in \{1,2,\ldots,k\},$$

where $d_{i,j}$ is denoting the initial dissimilarity between the data ω_i and ω_j and n_r the number of pairs in Γ_r.

De Soete (1984), following the ideas of Carroll and Pruzansky (1980), sets the problem up in a different way. He posed the problem as a mathematical programming problem. The objective function of this mathematical programming problem was composed of two terms, one measuring the difference between the dissimilarity of the dataset and the proposed distance and the other measuring (or penalising) how far the proposed distance was from being ultrametric. The last part depends on a parameter whose value increases with the steps of the algorithm. The first part of the function can be written as

$$L(U) = \sum_{i<j} (d_{i,j} - \delta_{i,j})^2$$

and the second part

$$P(U) = \sum_{\Lambda} (\delta_{i,k} - \delta_{j,k})^2,$$

where Λ is the set of triples that violate the ultrametric property:

$$\Lambda = \{(i,j,k)\,|\,\delta_{i,j} \leqslant \min(\delta_{i,k}, \delta_{j,k}) \text{ and } \delta_{i,k} \neq \delta_{j,k}\}.$$

Therefore, the function to minimise is

$$\Phi(U,\rho) = L(U) + \rho P(U) \quad (\rho > 0).$$

At each step of the algorithm a quadratic programming problem without constraints was solved and the value of the parameter ρ was increased. This approximation, even faster than the branch-and-bound algorithm, does not always guarantee the finding of the global optimum.

A comparison between both methods can be found in (Chandon and De Soete, 1984).

More recent work on this topic has been by Hubert and Arabie (1995). The authors modified, in a heuristic way, an iterative projection algorithm in convex sets for least squares problems, in such a way that the convex sets (in our case the inequalities) can be chosen in the execution of the algorithm. This algorithm not only finds least squares ultrametric distances, but tree distance or more general structures. Again this is a heuristic algorithm where the global optimum is not guaranteed.

A couple of papers related to this problem are by Awargala et al. (1995) and Gascuel and Levy (1996). In the former, an algorithm is proposed to look for the best ultrametric distance to a dissimilarity matrix; the norm used was L_1. On the other hand, Gascuel and Levy try to find the least squares tree distance of a set of dissimilarities.

3. Our genetic-based algorithm

The problem we want to solve is, given a dissimilarity matrix $D = [d_{i,j}]_{i,j=1,2,\dots,n}$ of a dataset, to find the ultrametric distance matrix $U = [\delta_{i,j}]_{i,j=1,2,\dots,n}$ that minimises the following formula:

$$\sum_{i<j}(d_{i,j} - \delta_{i,j})^2.$$

To solve this problem we have developed a new genetic-based algorithm. We call it the order-based approach. We have based this approach mainly on two ideas. First of all, we know that given an ultrametric distance U on a dataset Ω, it is possible to give a Θ order to the dataset Ω objects, with the following property: if the rows and columns of the U matrix are ordered according to Θ, then this new matrix will comply with the following properties (Lerman, 1981):

1. $\forall i \leqslant j \Rightarrow \delta_{i,j} \leqslant \delta_{i,j+1}$,
2. $\forall k \in \{1,2,\dots,n\}$ if $\delta_{k,k+1} = \delta_{k,k+2} = \cdots = \delta_{k,k+s+1} < \delta_{k,k+s+2}$, then
   - $\delta_{k+1,j} \leqslant \delta_{k,j}\ \forall k+1 < j \leqslant k+s+1$,
   - $\delta_{k+1,j} = \delta_{k,j}\ \forall j > k+s+1$.

Therefore, given Θ, an order for the objects of the dataset, and an n-tree structure T, it is then an easy task to find the least squares ultrametric distance of the dissimilarity of the dataset for this order and n-tree structure. The n-tree structure gives the entries of the ultrametric distance that are equal. Therefore, given the order and the tree structure, the problem can be set up as a quadratic programming problem with constraints in $n-1$ variables.

We can see this with an example. Let us suppose that Ω has 5 elements and the dissimilarity matrix of the dataset in the Θ order is

$$\begin{pmatrix} 0 & 4 & 3 & 5 & 1 \\ & 0 & 2 & 8 & 3 \\ & & 0 & 6 & 2 \\ & & & 0 & 9 \\ & & & & 0 \end{pmatrix}.$$

If the tree structure is given in Fig. 1, then the ultrametric matrix has the following pattern:

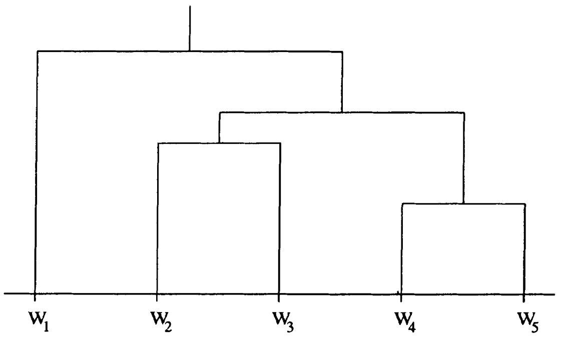

Fig. 1. An example of a n-tree structure.

$$\begin{pmatrix} 0 & \delta_1 & \delta_1 & \delta_1 & \delta_1 \\ & 0 & \delta_2 & \delta_3 & \delta_3 \\ & & 0 & \delta_3 & \delta_3 \\ & & & 0 & \delta_4 \\ & & & & 0 \end{pmatrix}$$

(we have eliminated in this case the double subscripts in order to emphasise the equal elements) and the quadratic programming problem can be written as follows:

$$\min\left(\left(\delta_1 - \frac{(4+3+5+1)}{4}\right)^2 + (\delta_2 - 2)^2 + \left(\delta_3 - \frac{(8+3+6+2)}{4}\right)^2 + (\delta_4 - 9)^2\right)$$

subject to the following constraints:

$$\delta_3 - \delta_1 \leqslant 0, \quad \delta_2 - \delta_3 \leqslant 0, \quad \delta_4 - \delta_3 \leqslant 0,$$
$$\delta_i \in \mathbb{R} \text{ and } \delta_i \geqslant 0, \quad i = 1,2,3,4. \tag{1}$$

Hence, we are carrying out a search in the set of orders Θ and n-tree structures T. For that an individual of our order-based GA is a permutation of length $n + (n-2)$ of the numbers $1,2,\dots,2n-2$. The individual codifies an order for the dataset and an n-tree structure. The numbers from 1 to n codify the order for the elements of the dataset, and the numbers from $n+1$ to $2n-2$ codify the n-tree structure. We know that this representation is not unique. There exist different combinations of n-tree structures and orders in the data that produce the same ultrametric distance. In fact, an equivalence relation could be set up in the set of pairs (order, tree-structure). It does not damage the search capabilities of the GA (Gerrits and Hogeweg, 1990).

The result given by Chandon et al. (1980), and mentioned above, is the second idea we have based our approach on. This result suggests using the following heuristic to save computational time. In our algorithm, instead of solving a quadratic programming problem at every step, which is computationally expensive, we have approximated each variable by the mean of the corresponding values of the dissimilarities of the dataset. In the example, the values are

$$\delta_1 = \frac{(4+3+5+1)}{4} = 3.25, \quad \delta_2 = 2,$$

$$\delta_3 = \frac{(8+3+6+2)}{4} = 4.75, \quad \delta_4 = 9.$$

In case the resulting distance is not an ultrametric distance, as it is in the example, the values of some of the variables would be changed in order to obtain ultrametricity. We fix the value of the variable that must have the highest value, in our case δ_1. The values of rest of the variables change (if necessary) to satisfy the constraints (1) and consequently the ultrametric conditions. In the example, the following values are finally obtained:

$$\delta_1 = 3.25, \quad \delta_2 = 2,$$

$$\delta_3 = 3.25, \quad \delta_4 = 3.25.$$

As we can see only the variables δ_3 and δ_4 have changed their value.

The advantage of evaluating an individual in this way consists of spending less time than if we have to solve a quadratic programming problem in each evaluation of an individual. At the same time, following the Chandon et al. (1980) result, previously mentioned, if the order and n-tree structure are the optimum, then we will get the optimal ultrametric distance.

4. Experiments

We have carried out some experiments with four datasets. The first was taken from Shepard et al. (1975). It is a set representing the dissimilarities among the first 10 single-digit integers $\{0, 1, \ldots, 9\}$ considered as abstract concepts. In the second, taken from Rao (1952), the dissimilarities between 12 Indian casts and tribes are represented. The third taken from Kaufman and Rousseeuw (1990, p. 57) is a set of size 18 with dissimilarities among garden flowers. Finally, the well-known Ruspini dataset, a set of size 75, was taken as the fourth.

We used a steady-state GA (Whitley and Kauth, 1988). Four crossover operators: PMX, CX, OX1 and AP, in combination with six mutation operators: SM, SIM, ISM, IVM, EM and DM (Larrañaga et al., 1999) were used. The mutation probability and the size of the population were set in relation with the length of the individual. If we call l the length of the individual, the probability of mutation was set to $1/l$ and the size of the population was set to $3 \times l$. We used two different stop rules. If the best individual of the population did not change its value in 50 000 generations we stopped the algorithm, and if the number of generations was bigger than a fixed value it was also stopped. These values depending on the size of the sets were 80 000, 110 000, 150 000 and 300 000, respectively.

We carried out 10 experiments for each dataset, and each combination of crossover and mutation probabilities, i.e., for each dataset we carried out 240 experiments, 60 for each crossover operator and 40 for each mutation operator.

To compare our order-based GA we applied three more algorithms to these sets. The branch-and-bound algorithm (B&B) by Chandon et al. (1980), the mathematical programming algorithm (MPA) by De Soete (1984), and finally the iterative projection method (IPM) by Hubert and Arabie (1995).

The B&B algorithm could only be applied to the first two datasets. In the following two datasets a large amount of computing time was needed which made its application impossible (in the Kaufman and Rousseeuw set, it took seven days to explore the first branch). On the other hand, MPA and IPM could not be applied to the Ruspini dataset, owing to the limited computer memory resources (it exceeds the maximum memory limits of our compiler stack) in the first algorithm, and because of the version that we have, it could only work with datasets with less than 51 observations in the second algorithm. In each of the first three datasets MPA and IPM were applied 60 times.

All the experiments were carried out with a Pentium 200 MHz with 64 MB of RAM.

The goodness-of-fit index used is the percentage of the variance of dissimilarities that is accounted for by the ultrametric distance, denoted by VAF (which equals the cophenetic correlation coefficient):

$$\mathrm{VAF} = 1 - \frac{\sum_{i<j}(\delta_{i,j} - d_{i,j})^2}{\sum_{i<j}(\delta_{i,j} - \overline{\delta})^2},$$

where $\delta_{i,j}$ and $d_{i,j}$ are the initial dissimilarity and the distance between data i and j, and $\overline{\delta}$ is the mean dissimilarity value of the dataset.

Table 1 summarises the max, mean and min VAF values and the mean execution time in seconds obtained in the first three datasets with the algorithms MPA and IPM. In the case of the B&B algorithm the values belong obviously to just one execution.

In Table 2 the mean VAF values and mean execution time of the different crossover and mutation operators of the order-based GA in the application to the three first datasets can be seen.

We can assume from the results that the IPM algorithm and our order-based GA with the CX crossover operator have the best behaviour. Our order-based GA with CX outperforms the results obtained by the IPM in the three datasets whereas IPM has a better mean execution time than the order-based GA. Not only does the order-based GA algorithm obtain the optimum more times but it also has a better mean. On the other hand the IPM algorithm obtains lower variance than the order-based GA.

In relation to Shepard dataset the order-based GA with CX obtained the optimum VAF in 34 executions out of the 60 whereas the IPM

Table 1
Results of the experiments with the B&B, MPA and IPM algorithms

	Shepard (10)				Rao (12)				Kaufman (18)			
	VAF				VAF				VAF			
	Max	Mean	Min	Time	Max	Mean	Min	Time	Max	Mean	Min	Time
B & B	0.494	–	–	248.2	0.572	–	–	23633	–	–	–	–
MPA	0.406	0.403	0.395	16.40	0.530	0.525	0.505	49.51	0.450	0.436	0.414	689
IPM	0.494	0.481	0.457	5.71	0.561	0.538	0.495	8.29	0.511	0.492	0.447	15.4

Table 2
Results of the order-based GA in the three first datasets

	Shepard (10)				Rao (12)				Kaufman (18)			
	VAF				VAF				VAF			
	Max	Mean	Min	Time	Max	Mean	Min	Time	Max	Mean	Min	Time
CX	0.494	0.491	0.347	7.967	0.561	0.552	0.411	15.122	0.511	0.502	0.278	33.65
OX1	0.494	0.460	0.349	8.414	0.561	0.499	0.335	13.541	0.510	0.462	0.299	31.66
AP	0.494	0.450	0.347	8.403	0.561	0.508	0.411	14.372	0.504	0.395	0.278	33.17
PMX	0.494	0.460	0.399	8.494	0.561	0.510	0.411	13.815	0.510	0.429	0.319	47.04
SM	0.494	0.452	0.366	8.290	0.561	0.498	0.335	13.569	0.511	0.437	0.278	28.53
SIM	0.494	0.463	0.404	8.028	0.561	0.522	0.411	12.426	0.510	0.438	0.302	31.31
ISM	0.494	0.452	0.347	7.672	0.561	0.501	0.414	14.186	0.510	0.428	0.294	26.08
IVM	0.494	0.471	0.349	9.007	0.561	0.526	0.430	15.456	0.511	0.458	0.288	27.37
EM	0.494	0.481	0.403	8.714	0.561	0.528	0.463	14.463	0.511	0.467	0.348	44.95
DM	0.494	0.473	0.397	8.206	0.561	0.528	0.468	15.175	0.510	0.454	0.321	40.04

Table 3
Results of the experiments with order-based GA and the Ruspini dataset

	Ruspini (75)			
	VAF			
	Max	Mean	Min	Time
CX	0.851	0.494	0.097	1568
OX1	0.851	0.500	0.336	1532
AP	0.851	0.208	0.097	1540
PMX	0.851	0.456	0.194	1529
SM	0.851	0.392	0.129	1542
SIM	0.851	0.437	0.137	1521
ISM	0.851	0.383	0.101	1515
IVM	0.851	0.411	0.097	1549
EM	0.851	0.453	0.138	1587
DM	0.851	0.412	0.011	1542

algorithm only took the optimum in 14 out of the 60 executions.

In relation to Rao dataset the best value achieved for both algorithms was 0.561 that was taken by the IPM algorithm 25 out of 60 times and for the order-based GA with CX 26 out of 60 times.

Finally in Kaufman dataset both algorithms got the same best result 0.5114, this result was taken by IPM 6 times and 7 times by the order-based GA with CX.

In relation to the Ruspini dataset, we could only apply the order-based GA approximation. The results on this dataset can be seen in Table 3.

As observed from Table 3, the CX operator is not the best in this case. This is because we stopped the algorithms when they had not even converged. In almost all the cases the algorithms used the maximum number of iterations, in the Ruspini dataset 300 000.

5. Conclusions

A new genetic-based algorithm is presented to search for the closest ultrametric distance, in a least squares sense, to the dissimilarity of a dataset. The algorithm can be applied to larger numbers of observations than have been reported on in the relevant literature to date. The results obtained in the experiments with the order-based GA are particularly favourable with the CX crossover operator (except in the Ruspini dataset) outperforming those obtained by other methods.

Acknowledgements

This work was supported by the Basque Country Government, Departamento de Educación Universidades e Investigación PI 96/12 grant. We thank P. Arabie, J.L. Chandon and T.J. Smith for providing some software and help and to the anonymous reviewers for their most helpful comments.

References

Awargala, R., Bafina, V., Farach, M., Narayanan, B., Paterson, M., Thorup, M., 1995. On the approximability of numerical taxonomy. Technical Report 95-46, DIMACS, Rutgers University, Piscataway, NJ 08855, USA.

Carroll, J.D., Pruzansky, S., 1975. Fitting of hierarchical tree structures, US-Japan Seminar on Multidimensional Scaling, University of California, San Diego.

Carroll, J.D., Pruzansky, S., 1980. Discrete and hybrid scaling models. In: Lenterman, E.D., Feger, H. (Eds.), Similarity and Choice. Huber, Bern, pp. 108–139.

Chandon, J.L., De Soete, G., 1984. Fitting a least squares ultrametric to dissimilarity data: approximation versus optimization. In: Diday, E. et al. (Eds.), Data Analysis and Informatics, III. North-Holland, Amsterdam, pp. 213–221.

Chandon, J.L., Pinson, S., 1980. Analyse Typologique Théories et Applications. Masson, Paris.

Chandon, J.L., Lemaire, J., Pouget, J., 1980. Construction de l'ultramétrique la plus proche d'une dissimilarité au sens des moindres carrés. RAIRO Recherche Operationnelle 14, 157–170.

De Soete, G., 1984. A least squares algorithm for fitting an ultrametric tree to a dissimilarity matrix. Pattern Recognition Letters 2, 133–137.

Gascuel, O., Levy, D., 1996. A reduction algorithm for approximating a nonmetric dissimilarity by a tree distance. J. Classification 13 (1), 129–156.

Gerrits, M., Hogeweg, P., 1990. Redundant coding of an NP-complete problem allows effective genetic algorithm search. In: Schwefel, H.-P., Männer, R. (Eds.), Parallel Problem Solving from Nature. Springer, Dortmund, pp. 70–74.

Goldberg, D.E., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading, MA.

Gordon, A.D., 1991. Classification: Methods for the Exploratory Analysis of Multivariate Data. Chapman and Hall, London.

Gordon, A.D., 1996. Hierarchical clustering. In: Arabie, P., Hubert, L.J., De Soete, G. (Eds.), Clustering and Classification. World Scientific Publishers, Singapore.

Hartigan, J.A., 1967. Representation of dissimilarity matrices by trees. J. Amer. Statist. Ass. 62, 1140–1158.

Hartigan, J.A., 1985. Statistical theory in clustering. J. Classification 2, 63–76.

Holland, J., 1975. Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor.

Hubert, L., Arabie, P., 1995. Iterative projection strategies for the least-squares fitting of the tree structures to proximity data. Br. J. Math. Statist. Psychology 84, 281–317.

Jain, A.K., Dubes, R.C., 1988. Algorithms for Clustering Data. Prentice Hall, New Jersey.

Jardine, C.J., Jardine, N., Sibson, R., 1967. The structure and construction of taxonomic hierarchies. Math. Biosci. 1, 171–179.

Johnson, S.C., 1967. Hierarchical clustering schemes. Psychometrika 32, 241–254.

Kaufman, L., Rousseeuw, P.J., 1990. Finding Groups in Data. An Introduction to Cluster Analysis. Wiley, New York.

Larrañaga, P., Kuijpers, C.M.H., Murga, R.H., Inza, I., Dizdarevic, S., 1999. Genetic algorithms for the travelling salesman problem: A review of representations and operators. Artificial Intelligence Review, to appear.

Lerman, I.C., 1981. Classification et Analyse Ordinale des Données. Dunod, Paris.

Rao, C.R., 1952. Advanced Statistical Methods in Biometric Research. Wiley, New York.

Shepard, R.N., Kilpatric, D.W., Cunningham, J.P., 1975. The internal representation of numbers. Cognitive Psychology 7, 82–138.

Whitley, D., Kauth, J., 1988. Genitor: A different genetic algorithm. In: Proceedings of the Rocky Mountain Conference on Artificial Intelligence, Vol. 2. pp. 189–214.