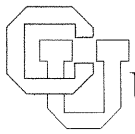
Computer Techniques for Cluster Analysis *

R. Michael Perry

CU-CS-244-83



University of Colorado at Boulder
DEPARTMENT OF COMPUTER SCIENCE

* This research was supported by ONR grant number N00014-78-C-0433.

COMPUTER TECHNIQUES FOR CLUSTER ANALYSIS

by

R. Michael Perry*

CU-CS-244-83 April, 1983

*Department of Computer Science, University of Colorado, Boulder, CO 80309

This research was supported by ONR grant number N00014-78-C-0433.

Computer Techniques for Cluster Analysis

R. Michael Perry

Abstract

Cluster analysis is a method for understanding the spatial arrangement of pointlike objects. It is practiced informally when stars are seen as forming galaxies or grains in film are viewed as depicting an image. This paper describes some computer techniques for cluster analysis of a set of points when distances between the points are known. In general points that are close together will be grouped in the same cluster. Moreover, clusters of points can be treated as single points and grouped into higher-order clusters, thereby obtaining a hierarchical arrangement that depicts large-scale features and fine detail as well. These techniques have been used in psychological studies of conceptualization and memory retention at the University of Colorado, and the applications are briefly reviewed.

1. Introduction

One of the basic mechanisms for understanding the world is to search for groupings or arrangements among objects that are distinguished primarily by spatial location. Thus stars are seen as forming constellations, clusters and galaxies. Grains in film similarly reveal an image even when they can be seen individually. Much can be learned by the recognition of groupings or clusters among pointlike objects, and techniques for automating this process thus are of interest.

This paper describes some computer techniques for cluster analysis of a set of points when distances between the points are known. These techniques were developed in connection with studies of human conceptualization and memory retention at the University of Colorado. The algorithms and computer implementation are described and their usage in the psychological studies is briefly reviewed.

2. Theory

In general in a clustering problem we are given a finite, nonempty set of points S with an implied distance relation, and are asked to find a "clustering" under which points that are close together will tend to be in the same grouping or cluster. Thus the mutual distances are regarded as defining a spatial arrangement of the points. The purpose of clustering then is to furnish an unambiguous interpretation of the structure of the spatial arrangement. The interpretation in turn will depend on the method chosen for clustering, but it can be hoped that significant features will not depend strongly on the method that is used.

Thus in particular the notion of "cluster" implies an arrangement or structuring of an "underlying set" of points chosen from S, into a set containing individual points or constituent clusters, with the additional requirement that the underlying sets of different constituent clusters must be disjoint. For formal purposes we define clusters over S inductively as follows.

1. Individual points $p \in S$ are clusters. The underlying set of a point p is taken to be $\{p\}$, and p is then said to be an *atomic* cluster and to have order 0. All other clusters will be nonatomic and will have order >0.

2. A nonatomic cluster C will have the form of a nonempty, nonsingleton set of clusters of lower order, which in turn will be called its *constituents*. The one additional requirement will be that the underlying sets of these constituents must all be disjoint. The clusters themselves will then be said to be disjoint. The order of C is defined as 1 + the maximum order of its constituents. The underlying set of C is defined as the union of the underlying sets of its constituents.

3. All of the clusters over S are obtained by applying rules (1) and (2).

In summary, (1) individual points are clusters, (2) any nonempty, nonsingleton set of clusters whose underlying sets are disjoint, is a cluster, and (3) these are the only clusters. Thus a cluster is a tree with the branches unordered. In particular there are only finitely many clusters over S. If S is a singleton then the only cluster is the single member of S (but not S itself). For larger S, S itself will be a cluster over S, in addition to its individual points. If $S = \{1, 2, 3\}$ the clusters over S are 1, 2, 3, $\{1, 2\}$, $\{1, 3\}$, $\{2, 3\}$, $\{1, 2, 3\}$, $\{\{1, 2\}, 3\}$, $\{\{1, 3\}, 2\}$, $\{1, \{2, 3\}\}$. For larger sets there are many more clusters.

Given a cluster C over S, a *subcluster* is defined inductively as either C, or if C is nonatomic, one of its constituents or a subcluster of one of its constituents. Thus in view of assumption (2) the underlying sets of two subclusters of C must be disjoint, or one underlying set must be included in the other.

A *clustering* of S is a cluster over S whose underlying set is S. Thus in the above illustration there are four clusterings of the three-element set S, and in addition, six other clusters over S. Although many clusterings of any sizable S are possible, the meaningful ones, from our point of view, are those in which objects that are close together are grouped into clusters of low order. Moreover, to form higher-order clusters in a meaningful way it is necessary to extend the measure of distance, assumed to be given for the individual points, to arbitrary clusters. In this way the entire set S can be formed into a cluster that reveals large-scale structure as well as fine detail in the spatial arrangement of points.

Something should be said about the allowable measures of distance, whether between points or clusters. In the most restrictive case the distance function d was required to be a metric, that is to satisfy the following properties for points (or clusters) x, y and z.

1. $d(x,y) \geq 0$, with equality holding if and only if $x = y$.
2. $d(x,y) = d(y,x)$.
3. $d(x,z) \leq d(x,y)+d(y,z)$.

However these properties were not always enforced, particularly in the case of nonatomic clusters. The minimum conditions that were always enforced were:

1. $d(x,y) \geq 0$, with equality holding if $x = y$.
2. $d(x,y) = d(y,x)$.

That is, the distance was always required to be nonnegative and reflexive, or independent of the direction of measurement.

The methods of clustering of interest here, then, are *distance-based*. There are two main steps in formulating a method of this type: (1) establishing the distance-based criterion under which two objects (whether points or clusters) will always be placed in the same cluster, and (2) extending the measure of distance as far as necessary, so that a distance is defined between any two clusters that would be considered for inclusion in a cluster of higher order.

Clustering then proceeds from the original, unstructured set of points. Points that meet the necessary criterion are grouped into clusters. One such cluster may contain more than two points because clustering is transitive, that is, if a and b are in the same cluster, and b and c are in the same cluster, then a and c must be in the same cluster. In general neighboring points will be grouped together. On the other hand a cluster may consist of a single point if no suitable neighbor can be found. In any case, the points are grouped into clusters as far as the criterion allows, then the process is iterated, clusters being grouped into clusters of higher order. In this manner the entire set of points is finally structured into one large cluster. Thus on each iterative step, if there are at least two objects, that is, if the final cluster has not already been reached, at least two objects must be grouped into a cluster.

3. Methods for Distance-Based Clustering

In the work reported here there were two main criteria for placing objects in clusters during one iterative step of analysis. In the first version two objects were put in the same cluster if either object was a nearest neighbor of the other. In the second version the two objects had to be *mutual* nearest neighbors to be guaranteed placement in the same cluster. In either case the distance measure had to be extended to higher-order clusters. The main means of doing this was to define the distance between two clusters as the *separation distance* between the underlying sets, that is, the minimum distance from a point in one underlying set to a point in the other.

Other measures than the separation distance, which is not a metric, were used on occasion. It was found, however, that even when the properties of a metric were enforced the results were not much affected so that the separation distance, which is easy to compute, became standard.

The methods of clustering, then, were primarily distinguished by how they placed objects in the same cluster during one step of analysis. The first version will be referred to as the *nearest-neighbor* method and the second (for reasons given later) as *contouring*. Both methods are illustrated in fig. 1, in which a hypothetical array of five points is to be clustered based on the usual Euclidean distance in the plane. This initial array is shown with the points labeled in (1a). The successive steps for the nearest-neighbor method are shown in (1b-c) and the steps for contouring are shown in (1d-f).

Thus the nearest-neighbor method takes only two iterative steps to complete the clustering, while three are needed for contouring. Moreover on each step of the first method any cluster is paired with at least one other one (since it must have a nearest neighbor), thus all constituents of a cluster must have the same order. Contouring, however, allows constituents to have differing order. Thus in the clustering of (1c) every constituent has order 1 while in that of (1f) the two constituents have orders 0 and 2.

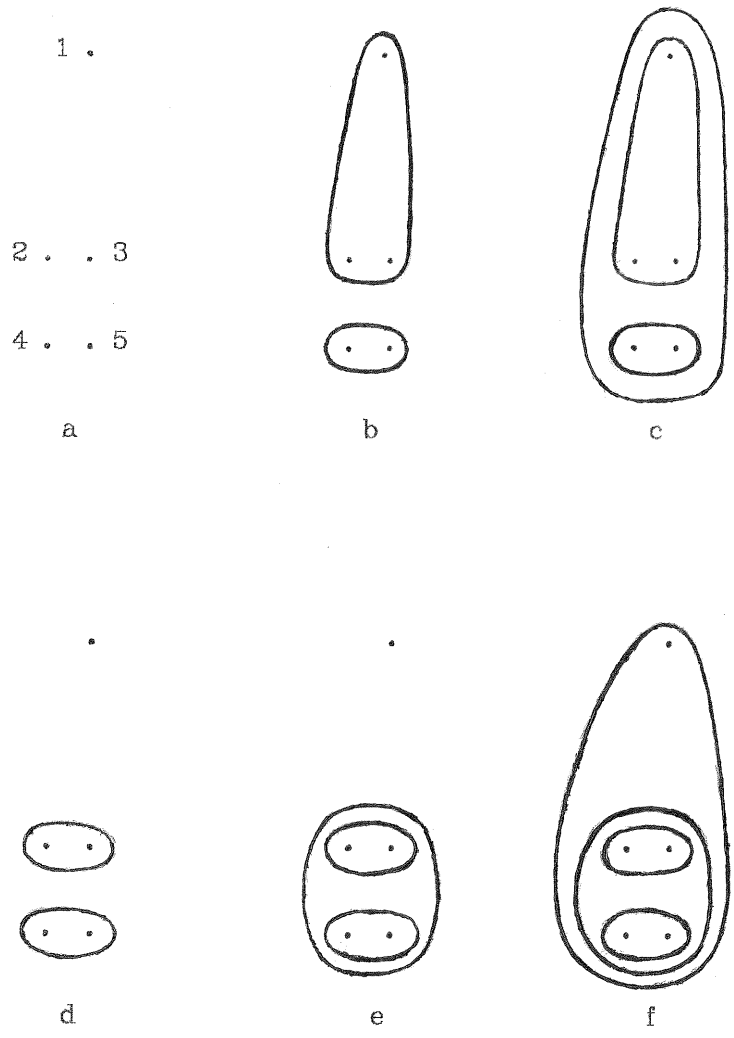


Fig. 1, showing the successive stages of clustering for two different methods. (a) original array of points; (b-c) nearest-neighbor method; (d-f) contouring.

The two clusterings, moreover, show significant differences in structure. Thus in (1c) point 1 is paired with points 2 and 3 in one constituent while in (1f) it is isolated as a constituent by itself. In (1c) on the other hand, points 4 and 5 form one constituent while in (1f) two clusters consisting respectively of points 2 and 3 and points 4 and 5, are formed into one constituent. All this is a consequence of the fact that point 1 is a considerable distance from the other points, even its nearest neighbor, point 3. Thus the first method is forced to group point 1 with point 3, regardless of the distance, while the second method, by isolating point 1 and forming the other points into one constituent, is able to give a better indication of the actual distances involved in the spatial arrangement.

In fact for the second method each cluster of any order has a "dispersion distance" such that (1) the constituents are within this distance of their nearest neighbors which are also within the cluster and (2) any other, disjoint cluster must be at a greater distance from any constituent and thus from the cluster as a whole. The given cluster, then, is contained in a "contour" drawn at the dispersion distance around its constituents, and thereby is isolated from all other disjoint clusters; thus the method has been referred to as "contouring". In general contouring yields a more detailed structure that better reflects the large disparity that may exist among the nearest-neighbor distances. Often, however, there is a great deal of fine structure for this method so that some "coarsening" -- removal of contours -- is helpful in visualizing the larger-scale structure.

4. Algorithms

Both methods of clustering can be carried out efficiently, that is, in time that is polynomial in the number of points in the set, $|S|$, with reasonable assumptions about the representation of the points in S and the difficulty of computing distances between them. On each iterative step of clustering we must determine the distance between clusters; clearly this is a polynomial-time operation since it depends only on the distances between points in the underlying sets, which are disjoint subsets of S. (A polynomial-time clustering was also obtained with other definitions of the inter-cluster distance.) For the nearest-neighbor method any cluster is joined with at least one other one on each iterative step; thus the number of clusters is reduced by at least half and the total number of steps therefore is not more than $\log_2|S|$. With contouring it is possible for only two clusters to be joined on each step so the number of steps could be as large as $|S|-1$, but the timing will still be polynomial in $|S|$. (In practice it has not been excessive compared with the other method.)

An important feature of many of the clustering problems to date is that many of the point-to-point distances are infinite (in practice, a very large number). Thus a boolean relation is given such that points are a finite distance apart or are "connected" only if the relation holds between them. Typically the boolean connections are not much more numerous than the points themselves but they always form a connected graph, so that a path of connections can be formed between any two points. This means that any reasonable distance-based clustering will have all constituents of any subcluster at finite distances from their nearest neighbors.

The nearest-neighbor method is implemented as follows. On each iterative step an array of clusters is formed. These clusters are to be grouped into clusters of the next-higher order. To each cluster in the array is associated its "adjacency list" -- those clusters at a finite distance from it. These clusters are sorted by increasing distance and those at the smallest distance -- the nearest neighbors -- are "marked" along with the original cluster, with a number denoting the cluster of next-higher order into which they will be placed. Meanwhile

the original cluster is placed on a list assigned to this number. This list, in effect, is the cluster of next-higher order that is being formed.

The marking itself proceeds by examining each cluster in the array in turn. If a cluster has already been marked with a number this number is retained; otherwise a new number is created. Then the nearest neighbors of the cluster are examined. All of these should be marked with the same number as the original cluster. For those that are, nothing is done. If any have not yet been marked at all, then they are assigned the new number and are added to the cluster list for this number. However, if a cluster has already been marked with a different number, then (1) the cluster list assigned to the old number is retrieved, (2) all clusters in this list, together with the neighboring cluster itself, are marked with the new number, (3) all these clusters are added to the cluster list for the new number, and (4) the cluster list for the old number is made empty. In this way clusters of the next-higher order are gradually built up and, whenever two clusters touch each other, they are coalesced into one.

When the process is complete the numbers denoting the higher-order clusters are examined. For those having nonempty lists the lists are retrieved -- these are the clusters of next-higher order. If there is only one such cluster the analysis is complete. Otherwise the iterative step is repeated with the new clusters.

For contouring we could use the same approach, that is, form the new clusters the same way on each step, except that now the neighboring clusters would have to be mutual nearest neighbors, in keeping with the rationale for contouring. Instead we use a different algorithm which sometimes is much faster, and appears to be faster in general. This algorithm is recursive rather than iterative. Initially it is given list of clusters to be formed into a clustering. If this list is a singleton, that is, if it has only one element, then just the one element is returned. If it contains two elements or is larger but all the nearest-neighbor distances are the same, then the list itself is returned. Otherwise a nontrivial clustering is carried out.

First clusters are formed into an array and to each is associated an adjacency list as before. Next a "contouring distance" is chosen. Currently this is the median of the nearest-neighbor distances of all clusters in the array (with a given distance being counted the number of times it occurs, rather than just once), but other choices would also be acceptable, as is noted later. Any two clusters that are not more than this distance apart are grouped together into a cluster of next-higher order.

The grouping into clusters is done the same way as in the nearest-neighbor method except that different clusters are treated as "nearest neighbors", in this case, those that are not more than the contouring distance from each other. Thus (1) not all clusters grouped together will be nearest neighbors and (2) not all nearest neighbors will be grouped together. A cluster that is more than the contouring distance from its nearest neighbor will remain isolated. When the grouping into clusters is complete, then, there will be some newly-formed clusters of higher order and some isolated clusters that were not grouped.

Next the algorithm is applied recursively to each newly-formed cluster in turn using its constituents as the initial list of clusters. In this way these clusters are structured as they will appear in the final clustering. Finally the algorithm is applied recursively again, this time to the list containing all the newly structured clusters, together with the older, isolated ones. This results in the final clustering.

In particular it is easy to show that, unless the algorithm is given a singleton list of clusters to start with, this case can never arise. Instead the

clustering will proceed recursively until each list to be clustered has all the nearest-neighbor distances the same and no further clustering is possible. For the algorithm to work properly, however, the contouring distance must be chosen so that clustering will go to completion. Any distance less than the maximum nearest-neighbor distance, and not less than the minimum such distance will do, though certain distances are expected to result in faster clustering.

A particular case that illustrates how timing depends on the method is one in which the points, call them $p_1, p_2, \ldots, p_n$, are structured so that their distances $d(p_i, p_{i+1})$ form a decreasing sequence of (finite) values while all distances $d(p_i, p_j)$ for $|i-j|>1$, are infinite. Thus p_{i+1} will be the (unique) nearest neighbor of p_i whenever $1 \leq i < n$ but only p_{n-1} and p_n will be mutual nearest neighbors. If these are formed into a (two-constituent) cluster then only it and p_{n-2} of the remaining points will be mutual nearest neighbors, and so on.

Thus if we applied the nonrecursive technique initially suggested for contouring, in which only the mutual nearest neighbors would be grouped on each iterative step, only one new cluster could be formed on each step and the number of steps needed would be $n-1$. Each of these in turn would require time proportional to the number of clusters so the overall timing would be proportional to n^2.

With the recursive algorithm, using the median of the nearest-neighbor distances as the contouring distance, the size of the problem is reduced by half for each of the two recursive calls that follow the initial call, so the timing will be proportional to $n\log_2 n$. The timing will vary if a different choice of the contouring distance is used. For example, suppose it is chosen to be the kth smallest of the nearest-neighbor distances, with k/n close to some constant c between 0 and 1. (For the median c would be 0.5.) The timing can be shown to be proportional to $n\log_b n$ where $b = 1/max\{c, 1-c\}$, so that the ratio of the timing to that for the median case is $\log_b n / \log_2 n$. This ratio, consequently the timing, is minimized for the median case in which b reaches its maximum of 2.

This result, however, is problem-dependent. For other distance relations on the n points other choices of contouring distances may give faster timings. Still the recursive algorithm, with some reasonable choice of the contouring distance, seems likely to have better worst-case behavior than the iterative algorithm.

Typically with contouring there is much fine structure and the order of the clustering is much higher than in the nearest-neighbor case. Thus it is desirable to show several versions of the clustering representing differing amounts of "coarsening" or removal of contours from the original clustering. This will make the larger structures more apparent and also will give some indication of how much the clustering depends on small differences in the point-to-point distances or how "robust" the clustering is. Thus if what is expected to be a small amount of coarsening in fact destroys most of the structure then it must have depended on small differences in the distances but if most of it survives then it is more robust.

The rationale used for coarsening is to start with the original set of n points, their nearest-neighbor distances, and the "fine-structure" clustering given by contouring. Next a contouring distance d is chosen as the kth smallest of the nearest-neighbor distances where k/n is close to a fixed value c. (Actually we choose k so that $(k-1)/(n-1)$ is as close as possible to c.) Next all the nonatomic subclusters in the clustering are examined "top-down" -- that is, beginning with the whole clustering, proceeding to its constituents, then to their constituents and so on. Any subclusters in which the dispersion distance is not more than d are "flattened" -- replaced by their underlying sets -- and are then

marked "atomic". From this point on they will be treated as single points and will not be further coalesced during more advanced stages of the coarsening.

For the next stage, then, the partially-coarsened clustering will be further coarsened using the same method. However the "points" will consist of the "top-level-atomic" clusters -- those that are atomic or are marked such and are not subclusters of any others marked atomic. There will be fewer such "points" than on the previous step. Their nearest-neighbor distances are determined and the contouring distance is selected as before, using the same value of c. Next the clustering is examined top-down again, and any subclusters that are not marked atomic and whose dispersion distance is not more than the contouring distance are flattened, down to the top-level-atomic subclusters, and are marked atomic themselves.

The above step is iterated until it can go no further, that is, until the contouring distance equals the dispersion distance of the entire clustering. In particular if $c=1$ then many contours will, in general, be removed leaving a coarse structure, while more of the fine structure will be preserved for smaller c, with all of it being retained if $c=0$.

Sample experimental output for both the nearest-neighbor method and contouring, including coarsening, will be shown in the next section. Both methods as implemented require extensive list processing as well as arrays of lists, so that LISP was found to be a convenient programming language. In particular the LISP recursion facility was useful for contouring because it allowed array names and the arrays themselves to be dynamically created and destroyed on recursive calls.

The programming was done on a VAX 11/780 computer in Franz LISP, under the UNIX operating system. Execution times varied widely depending on the paricular application. Typically the CPU time was of the order of 1-2 min. for a set of about 50 points with about 80 boolean connections, so that most of the point-to-point distances were infinite. Much more time, about 20-30 min., was needed for this number of points with every distance finite, since this made the adjacency lists much larger. No doubt the timing could be improved; so far this has not been necessary.

5. Applications

At present there are four major computer techniques for clustering, each in two versions, that is, using the nearest-neighbor method and contouring, making eight programs in all. All were created for studies of human conceptualization, learning and memory retention at the University of Colorado Department of Psychology [1]. In these studies model objects such as helicopters or cranes are assembled from component parts. Typically an object has about 50 parts. Each part is connected to one or more others. Parts that are not physically connected are considered to be an infinite distance apart, except for a few cases in which a finite distance implying a "connection" was accepted on grounds of symmetry. The number of connections in the object ranged from a few percent more to 50% more than the number of parts; thus most of the parts had only a few parts connecting.

In the most basic clustering problem we are given the order of request of the parts when an object is assembled and are asked to find the "conceptualization" used in making the assembly. Essentially the conceptualization is a hierarchical subdivision of the object in which the individual connected parts are grouped into larger connected units, and these into still larger units, and so on. In short it is a distance-based clustering of the parts. Parts that are connected

and whose orders of request are nearly equal are assumed to belong near each other in the clustering, that is, as part of the same subcluster of low order. The distance between connected parts, then, is just the absolute value of the difference in the orders of request. Thus the parts are treated as individual points and the clustering is formed. One variation of this analysis is to obtain a "consensus" conceptualization by averaging the orders of request over several assembly trials.

A second problem concerns the clusterings that are obtained by the above analysis when a number of subjects all assemble the same object under varying conditions. We would like to know if these clusterings fall into meaningful patterns or hierarchies. For instance we can ask whether there is essentially one conceptualization with minor variations or several essentially different conceptualizations. Thus a clustering of the conceptualizations is called for.

To do this we must define a distance between conceptualizations. The way this has been done is to consider the boolean relation or set of connections in the object. The connections, represented by pairs of parts, are numbered $1, 2, \ldots, m$. Each connection is represented by a pair $\{p_1, p_2\}$. Given any clustering there will exist a minimal subcluster C that contains both p_1 and p_2, so that p_1 and p_2 will be in different constituents of C.

Next we define the *order* of $\{p_1, p_2\}$ as the order of C. This is the "bottom-up" order since it depends on the order of the constituents of C. An alternative is the "top-down" order which is the depth of C within the clustering, where the clustering itself will have depth zero, its constituents depth one, their constituents depth two and so on. At any rate for any clustering we obtain an m-vector $(v_1, v_2, \ldots, v_m)$ where v_i is the order of the ith connection. It is not difficult to show that an entire clustering can be reconstructed from its associated vector. (For example we can start with the pairs of lowest bottom-up or highest top-down order and add pairs of succeeding orders until the entire arrangement of pairs is determined.) Thus two clusterings with the same vector must be identical.

Finally, the distance between two clusterings is defined as the distance between the associated vectors, using a metric on an m-dimensional vector space. The metric that has been most useful is the Euclidean 1-norm, that is, the sum of the absolute values of the differences of the corresponding terms of two vectors. The bottom-up order and the 1-norm distance were used extensively in studies based on the nearest-neighbor method of clustering [1]. For contouring, however, the top-down order is probably more appropriate since the clusterings are not "flat-bottomed"; that is, constituents in a subcluster can have variable order. This means that the bottom-up order of a connecting pair can depend on a constituent that contains neither point of the pair, something that does not occur with the top-down order. The latter, then, seems to offer a more reasonable indication of how "similar" the components of a connection are.

It should be noted that in this sort of analysis, regardless of how the order is measured, all distances between points (in this case, clusterings) are finite. Thus there are large adjacency lists for the points and extensive computation is required for a moderately-sized problem; about 25 min. of CPU time on the VAX computer was needed for one typical case involving 47 points.

Thus far two techniques for clustering problems have been described, both dealing with cases in which the objects were correctly assembled. Two other techniques have been developed for understanding what is involved when the assembly is incorrect. (Incorrect assembly occurred when objects were reconstructed from memory, without a correctly assembled object as reference.)

In the first case we are trying to determine how the different connections in the object influence the errors that are made. Thus the connections are treated as the "points" in a cluster analysis. Two such connections are connected themselves if they have an endpoint in common. Thus a boolean relation can be defined on the set of connections. Two connections that are not connected to each other are at an infinite distance apart; otherwise the distance will depend on how errors are made at these connections when assembly of the object is attempted.

Thus it is assumed that, whenever the object is incorrectly assembled, there is a way of unambiguously deciding at which connections the errors are made. For the analysis, then, a number of cases of an incorrectly-assembled object are considered. To define the distance between two connections in the object that have an endpoint in common we consider the sets S_1 and S_2 of cases in which errors were made on the first or second connection, respectively. The distance is then defined as $|S_1 - S_2| / |S_1 \cup S_2|$, that is, as the ratio between the number of points in the symmetric difference of the two sets (where the symmetric difference is the set of points in the union that are not in the intersection) and the number of points in the union. This distance can be shown to be a metric on finite, nonempty sets [2]. In particular the distance is always ≤ 1; it is set to 1 by default if $S_1 \cup S_2$ is empty.

The distance is smaller if there is greater agreement between the errors made on one connection and those on the other. Thus the clustering will tend to identify groups of connections for which the pattern of errors is similar, such as those on which nearly all subjects made errors. In this way it is hoped that connections that are consistently troublesome can be identified so that instructional sequences can be designed to reduce the incidence of errors.

The second technique, which is the final one considered here, is concerned with identifying meaningful patterns of error making among different subject groups. Subjects within a group assemble the object under similar conditions. For example, visual instructional material may be presented. For each subject within a group, then, we have the list of connections on which errors were made. For any group we define an m-vector $(v_1, v_2, \ldots, v_m)$ by v_i = average number of errors made on the ith connection = number of subjects who made the errors divided by the number of subjects in the group. The distance between groups is then defined as the distance between vectors, measured as before, and clustering can proceed as in the earlier case.

An illustration of this technique is shown in fig. 2. The twelve points in each clustering represent twelve subject groups. The object in this case has 58 connections. Thus each point represents a location in a 58-dimensional vector space and the point-to-point distances, which were given by the 1-norm, are only roughly indicated in the figure. (2a) shows the clustering obtained by the nearest-neighbor method, while (2b-d) show the results of contouring with increasing amounts of coarsening. Thus (2b) has no coarsening, (2c) has a moderate amount, with the coarsening parameter $c = 0.5$, and (2d) has the largest amount, with $c = 1$. Certainly the two methods show differences but there are basic similarities too. In particular the nearest-neighbor method seems to correspond best to contouring with a moderate amount of coarsening.

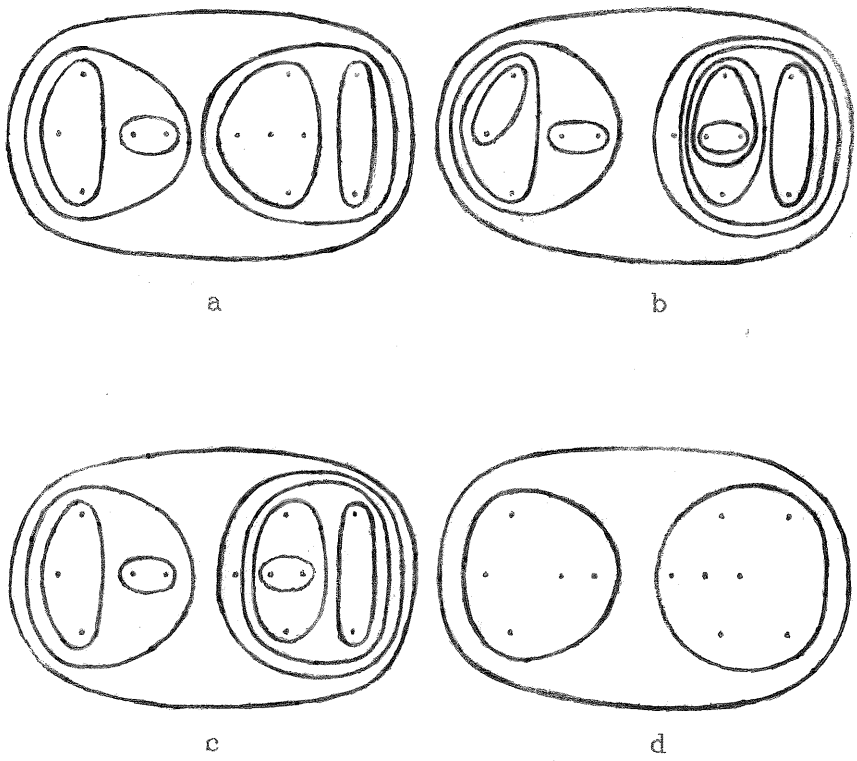


Fig 2, showing results of clustering by the nearest-neighbor method (2a) and contouring with increasing amounts of coarsening (2b-d).

REFERENCES

[1] Baggett, P. Four principles for designing instructions. Technical Report #121-ONR. To appear.

[2] Ehrenfeucht, A., private communication.