AUTOMATED CONSTRUCTION OF CLASSIFICATIONS CONCEPTUAL CLUSTERING VERSUS NUMERICAL TAXONOMY

by

Ryszard S. Michalski
Robert Stepp

IEEE Trans. on Pattern Analysis and Machine Learning, Volume PAMI-5, No. 4, July, 1983.

Automated Construction of Classifications: Conceptual Clustering Versus Numerical Taxonomy

RYSZARD S. MICHALSKI AND ROBERT E. STEPP

Abstract—A method for automated construction of classifications called *conceptual clustering* is described and compared to methods used in numerical taxonomy. This method arranges objects into classes representing certain descriptive concepts, rather than into classes defined solely by a similarity metric in some *a priori* defined attribute space. A specific form of the method is *conjunctive conceptual clustering*, in which descriptive concepts are conjunctive statements involving relations on selected object attributes and optimized according to an assumed global criterion of clustering quality. The method, implemented in program CLUSTER/2, is tested together with 18 numerical taxonomy methods on two exemplary problems: 1) a construction of a classification of popular microcomputers and 2) the reconstruction of a classification of selected plant disease categories. In both experiments, the majority of numerical taxonomy methods (14 out of 18) produced results which were difficult to interpret and seemed to be arbitrary. In contrast to this, the conceptual clustering method produced results that had a simple interpretation and corresponded well to solutions preferred by people.

Index Terms—Classification theory, cluster theory, conceptual clustering, data analysis, inductive inference, knowledge acquisition, learning from observation, learning without teacher, numerical taxonomy, pattern recognition, theory formation.

I. Introduction

CLUSTERING is usually viewed as a process of partitioning a collection of objects (measurements, observations, etc.) into groups of similar objects, according to some numerical measure of similarity. Such an approach to clustering raises two fundamental problems: "what kind of similarity measure should be used to cluster objects?" and "should the numerical similarity between objects be the only principle for constructing clusters?" These questions are discussed in this paper, and given answers that are substantially different from the ones given by traditional techniques.

In the area of cluster analysis and the closely related field of numerical taxonomy, the similarity between objects is typically determined by some proximity measure in a multidimensional space spanned over an *a priori* defined set of object attributes. The clusters are then defined as collections of elements (*points*) of the space whose intracluster proximities are high, and intercluster proximities are low. Research in cluster analysis has been, therefore, primarily concerned with devising various object proximity measures and developing efficient algorithms utilizing these measures. Surveys of these measures can be found in Sokal and Sneath [15], Anderberg [1], and Diday and Simon [2].

Such an approach to clustering has several significant limitations. One is that clusters determined as groups of objects that are "close" to each other in a fixed, *a priori* assumed attribute space may lack any simple conceptual interpretations. A reason for this is that a similarity measure typically considers all attributes with equal importance and thus makes no distinction between those that are more relevant and those which are less relevant or irrelevant. Consequently, if there is coincidental agreement between the values of a sufficient number of irrelevant attributes, objects that are different in some major ways may be classified as similar. The approach has no mechanism for selecting and evaluating attributes in the process of generating clusters. Neither is there any mechanism to generate new attributes which may be more adequate for clustering than those initially provided. Another important limitation of conventional methods is that they do not produce any conceptual description of the clusters. The problem of cluster interpretation is simply left to the data analyst. This is a serious drawback because data analysts are typically interested not only in determining clusters but also in formulating some meaningful descriptions of them.

Also, traditional techniques do not seem to be much concerned with the ways employed by humans in clustering objects. Observations of how people cluster objects indicate that they tend to formulate one or a few carefully selected attributes (out of very many possible attributes), and cluster objects on the basis of these attributes. Each cluster contains objects that are similar to each other in the sense that they score similarly on the "important" attributes. Thus, descriptions of such clusters can be formally expressed as logical conjunctions of relations on these attributes. Different clusters are expected to have descriptions with different values of the selected attributes. In short, people tend to cluster objects into categories characterized by nonintersecting conjunctive concepts.

This brings us to another related limitation of traditional methods: they do not take into consideration any "Gestalt" concepts or linguistic constructs people use in describing object collections. Such concepts may be characterizations of a configuration of objects such as *T*-shaped, *V*-shaped, etc.

The idea of clustering objects into categories described by

Manuscript received September 11, 1982; revised February 4, 1983. This work was supported in part by the National Science Foundation under Grant MCS-82-05166 and in part by the Office of Naval Research under Grant N00014-82-K-0186.

The authors are with the Department of Computer Science, University of Illinois, Urbana, IL 61801.

0162-8828/83/0700-0396$01.00

single concepts, specifically conjunctive concepts (hence *conjunctive conceptual clustering*), as well as a methodology for its computer implementation was first introduced in Michalski [7]. It was subsequently expanded in Michalski and Stepp [11], [12].

The purpose of this paper is to summarize the main ideas behind the conjunctive conceptual clustering method and to compare it to techniques of numerical taxonomy. The paper also describes the conjunctive conceptual clustering program CLUSTER/2 (which is a successor to the earlier program, CLUSTER/PAF) and then presents results of applying it and a numerical taxonomy program NUMTAX (implementing 18 different techniques) to two clustering problems.

II. Specification of a Clustering Problem

This section discusses various components of a clustering problem that must be specified by a data analyst before a computer-based clustering method can be used.

A. Objects to be Clustered and Their Attributes

Typically, objects to be clustered come from an experimental study of some phenomenon and are described by a specific set of attributes (variables) selected by the data analyst. The attributes may be measured on different scales, such as nominal, ordinal, interval, ratio, and absolute. In a simple case, one may only distinguish between qualitative attributes (measured on the nominal or ordinal scale) from quantitative attributes (measured on the remaining scales). The initial measurements are subject to problem-dependent transformations, which may reduce the precision of the quantitative attributes or replace subranges of their values by qualitative properties (e.g., a numerical size may be replaced by characterizations such as "small size," "medium size," or "large size"). The attributes selected by the data analyst are not always all relevant to the clustering problem. In conventional approaches, the selection of relevant attributes is treated as a separate preliminary step. In the conjunctive conceptual clustering method, attribute selection is performed simultaneously with the formation of clusters. The method selects those attributes which, from the viewpoint of assumed criteria, allow it to "simply" characterize the individual clusters in terms of available concepts.

B. The Principle for Grouping Objects into Clusters

Objects are grouped together by a clustering method according to some principle. The traditional principle for grouping objects into clusters utilizes some numerical measure of object similarity, usually a reciprocal of a distance measure. In conceptual clustering, objects are assembled into clusters that represent single concepts (linguistic terms or simple logical functions defined on such terms). In *conjunctive conceptual clustering*, described here, objects are grouped into clusters that are characterized by logical products of relations on selected object attributes, i.e., conjunctive concepts. These relations may also include disjunction of properties, but only if the disjunction involves values of the same attribute. This type of disjunction is called *internal disjunction* [6]. Conjunctive concepts with internal disjunction seem to represent typical human characterizations of object classes. An example of such a concept is: "objects that are small and red, with either a green or a blue spot." A definition and a discussion of conjunctive concepts with internal disjunction is given in Section II-D.

C. The Type of Cluster Structure

Given a collection of objects E, the goal of clustering is to divide the collection into certain meaningful subsets. Let E_1, $E_2, \cdots, E_k$ be such subsets (clusters) of E, and let α_i denote a description of subset E_i. In general, a description α_i is satisfied not only by objects in E_i, but also by some unobserved objects. Based on the relationships among the clusters and cluster descriptions, three different types of intercluster structures are commonly distinguished in the literature.

- *Partition Structure:* A set of clusters whose union is the set E, and whose descriptions are all disjoint (this implies that the clusters themselves are disjoint).
- *Overlapping Structure:* A set of clusters whose descriptions intersect.
- *Hierarchical Structure:* A multilevel hierarchy in which clusters at the first level represent a partition of the initial set E, and clusters at a lower level are elements of partitions of the parent clusters one level higher.

The method described here generates either a partition structure or a hierarchical structure of clusters. An overlapping structure can be generated by a method described by Stepp [16].

D. Cluster Representation Scheme

The purpose of a cluster representation scheme is to simply and generally characterize objects in a cluster. Conjunctive conceptual clustering uses two cluster representation schemes: a single representative object selected from a cluster, called the *seed* of the cluster, and a conjunctive statement that describes all objects in the cluster. This conjunctive statement, called a *logical complex*, is an expression in the variable valued logic system VL_1 [4], [5].

Suppose that $x_1, x_2, \cdots, x_n$ are variables selected to represent objects. We will assume that each variable $x_i, i \in \{1, 2, \cdots, n\}$, has an assigned *domain*, $\mathrm{DOM}(x_i)$, that specifies all possible values the variable can take for any object in the collection to be clustered. The number of such values is given by d_i. The domains are assumed to be finite, and represented generally as $\mathrm{DOM}(x_i) = \{0, 1, 2, \cdots, d_i - 1\}$. We distinguish between *nominal*, *linear*, and *structured* variables, whose domains are unordered, linearly ordered, and tree-ordered sets, respectively. Examples of nominal variables are color and blood-type. Examples of linear variables are rank, size, and quantity-of-something. An example of a structured variable is shape, whose values may be triangle, rectangle, pentagon, $\cdots$, or polygon, which represents a more general concept (the parent node of the nodes representing triangle, rectangle, etc. in the tree-structured domain).

The description space spanned by variables $x_1, x_2, \cdots, x_n$ is called the *event space*. Each point (*event*) in the space is a vector of specific values of variables $x_1, x_2, \cdots, x_n$. An event that is a description of some object in the collection to be clus-

tered is called an *observed event*. Other events are called *unobserved events*.

A *relational statement*[1] (or a *selector*) is a form

$[x_i \# R_i]$

where R_i, the *reference*, is a list of elements from the domain of variable x_i, linked by the *internal disjunction*, denoted by "V." # stands for the relational operator = or $\neq$.

The selector $[x_i = R_i]$ ($[x_i \neq R_i]$) is interpreted as "value of x_i is one of the elements of R_i" ("value of x_i is not an element of R_i"). In the case of linear variables, the notation of a selector can be simplified by using relational operators $\geqslant$, $>$, $<$, $\leqslant$, and a range operator ". . " as illustrated below. Here are a few examples of a selector, in which variables and their values are represented by linguistic terms:

[length > 2]	(length is greater than 2)
[color = blue V red]	(color is blue or red)
[size $\neq$ medium]	(size is not medium)
[weight = 2..5]	(weight is between 2 and 5, inclusively).

A logical product of selectors is called a *logical complex* (*l*-complex). A set of objects that satisfy all selectors in an *l*-complex is called an *s-complex* (set-complex). Thus, an *l*-complex is the description of an *s*-complex. For example, the *l*-complex:

[height = tall][color = blue V red][length > 2] &

[size $\neq$ medium][weight = 2 . . 5]

(the operation AND is denoted by the & or implied by the concatenation of selectors) describes those objects that are tall, blue or red, with length greater than 2, not medium size, and of weight 2 through 5. The set of all such objects constitutes the corresponding *s*-complex. The distinction between *l*- and *s*-complexes is used to permit the application of logical or set-theoretic operators, respectively, whichever is more convenient. When this distinction is unimportant, the term complex will be used without a prefix.

Not every collection of objects constitutes an *s*-complex, i.e., not every collection can be precisely described by an *l*-complex. It is, however, possible to describe every collection of objects by an *l*-complex, if the *l*-complex is also allowed to describe some additional objects (i.e., if it is permitted to be a generalized description of the collection). For example, events

e_1: (blue, large, round)

e_2: (red, medium, round)

can be described by the complex

[color = blue V red][size $\geqslant$ medium][shape = round].

This complex, however, also covers the unobserved events

e_3: (red, large, round)

e_4: (blue, medium, round)

[1]This form was first introduced in the variable-valued logic system one (VL_1) [5].

which are distinct from e_1 and e_2. The number of the unobserved events covered by a complex is called the *absolute sparseness* of the complex.

Given a set of complexes, the absolute sparseness of the set is defined as the sum of sparsenesses of complexes in it. In addition to the absolute sparseness, we also introduce another type of sparseness, the *projected sparseness* of a set of complexes. This type of sparseness is applicable only to a clustering which is a set of pairwise disjoint complexes. Since complexes are pairwise disjoint, there exist, for any pair of complexes, at least two selectors with the same variable and disjoint references. Variables involved in such selectors are called the *discriminant variables* of a clustering. The projected sparseness of a clustering is the sum of the sparsenesses determined in the event space spanned over just the discriminant variables. For example, if given two observed events e_1 and e_2 above, the two complexes

- [color = blue][size = large][shape = round]
- [color = red][size $\leqslant$ medium][shape = round V square]

have projected sparsenesses of 0 and 1, respectively, in the space spanned by the two discriminant variables color, and size (assuming that size takes only values small, medium, and large).

E. A Criterion of Clustering Quality

The problem of how to judge the quality of a clustering is difficult, and there seems to be no universal answer to it. One can, however, indicate two major criteria. The first is that descriptions formulated for clusters (classes) should be "simple," to make it easy to assign objects to classes and to differentiate between the classes. This criterion alone, however, could lead to trivial and arbitrary classifications. The second criterion is that class descriptions should "fit well" the actual data. To achieve a very precise "fit," however, a description may have to be complex. Consequently, the demands for simplicity and good fit are conflicting, and the solution is to find a balance between the two.

A number of other measures can be introduced for evaluating the quality of a clustering. CLUSTER/2 uses a combined measure which can include any of the following elementary criteria:

- the fit between the clustering and the events
- the simplicity
- the commonality
- the disjointness
- the discrimination index.

The *fit* between a clustering and the data is computed in two different ways, denoted as T and P. The T measure is the negative of the total (absolute) sparseness of the clustering, i.e., the negative of the sum of the absolute sparsenesses of complexes in the clustering. The P measure is the negative of the projected sparseness of the clustering. The reason for using the negative values is to increase the degree of match as the sparseness decreases.

The *simplicity* of a clustering is defined as the negative of its complexity, which is the sum of costs attributed to each selector present in the complexes. One possible selector cost func-

tion is based on the number of elements found in its reference list. Selectors with few reference elements are less complex than those with many elements, and therefore should have a smaller cost. A simple measure of complexity can be computed as the number of selectors contained in the complexes, i.e., using a constant selector cost function that gives each selector a cost of 1.

The *commonality* of a clustering is the total number of properties shared by the events in each of the clusters. The commonality is measured by finding the total number of selectors that appear in the complexes. This criterion is analogous to the traditional clustering criterion of maximizing the similiarity (e.g., number of shared properties) of events within a cluster.

The *disjointness* of a clustering is measured by the sum of the degrees of disjointness between every pair of complexes in the clustering. The degree of disjointness of a pair of complexes is the number of selectors in both complexes that involve the same variable and have reference values that do not intersect. For example, the pair of complexes

- [color = red][size = small ∨ medium][shape = circle]
- [color = blue][size = medium ∨ large]

has the degree of disjointness 2, because 2 of the 5 selectors do not intersect (nonintersecting selectors are underlined). This criterion promotes clusterings with classes having many differing properties, and is analogous to the criterion of requiring maximal distance between clusters, used in coventional methods of clustering.

The *discrimination index* of a clustering is the number of variables that singly discriminate among all the clusters, i.e., variables having different values in every cluster description.

The definitions of the above criteria are such that the increase of any criterion value improves the quality of the clustering. The relative influence of each criterion is specified using the *lexicographical evaluation functional* (LEF). The LEF is defined by a sequence of "criterion-tolerance" pairs (c_1, τ_1), $(c_2, \tau_2), \cdots$, where c_i is an elementary criterion selected from the above list, and τ_i is a "tolerance threshold" ($\tau \in [0 \cdots 100\%]$). In the first step, all clusterings are evaluated on the first criterion c_1, and those that score best or within the range defined by the threshold τ_1 are retained. Next, the retained clusterings are evaluated on criterion c_2 with threshold τ_2, similarly to the above. This process continues until either the set of retained clusterings is reduced to a singleton (the "best" clustering) or the sequence of criterion-tolerance pairs is exhausted. In the latter case, the retained clusterings have equivalent quality with respect to the given LEF, and any one may be chosen arbitrarily. The selection of elementary criteria, their ordering, and the specification of tolerances is made by a data analyst.

III. Method and Implementation

This section describes the algorithm for conjunctive conceptual clustering implemented in the program CLUSTER/2 (the successor to the program CLUSTER/PAF [11], [12]). The algorithm consists of a *clustering module* and a *hierarchy-building module*, which are described in Sections III-D and III-E, respectively. The operation REFUNION and two fundamental procedures, REDUSTAR and NID are first described in Sections III-A-C.

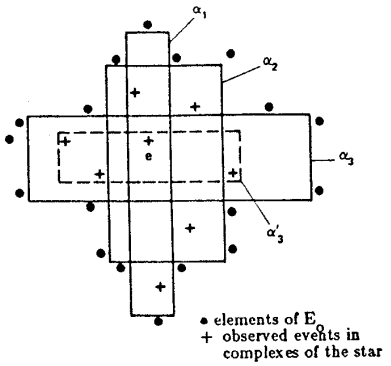


Fig. 1. An illustration of the star $G(e|E_0)$.

A. REFUNION Operation

This operation transforms a set of events and/or complexes into a single complex covering the events and/or complexes. For each variable, the set of all values the variable takes, in all given events and complexes, is determined. These sets are used as the reference of the variable in the generated complex. For example, given

e_1: (2, 3, 0, 1)

e_2: (0, 2, 1, 1) and

α: $[x_1 = 2..3][x_2 = 4][x_3 = 0][x_4 = 2]$

the refunion complex REFUNION(e_1, e_2, α) is

α': $[x_1 = 0 \vee 2 \vee 3][x_2 = 2 \vee 3 \vee 4][x_3 = 0 \vee 1][x_4 = 1 \vee 2]$.

The refunion complex has minimum sparseness (both absolute and projected) among all complexes covering the given events and/or complexes [7].

B. REDUSTAR Procedure

The *star* $G(e|E_0)$ of *event e against event set* E_0 $(e \notin E_0)$ is the set of all maximally general[2] complexes covering the event e and not covering any event in E_0. In other words, it is the set of all maximally general descriptions of event e which do not intersect with event set E_0. Fig. 1 presents a star of event e against events denoted by "•" in the two-dimensional space spanned over linear variables. The star consists of complexes α_1, α_2, and α_3. Complex α_3' is a "reduced" complex α_3, as explained below.

In the algorithm that follows, the "theoretical" stars defined above are subjected to two major modifications. The first is to minimize the sparseness of complexes in the stars, and the second is to "bound" the stars, i.e., to select from them a certain number of "best" complexes, according to a context-dependent criterion. The first modification is performed by

[2] A complex α is maximally general with respect to a property P if there does not exist a complex α^* with property P such that $\alpha \subset \alpha^*$.

algorithm REDUSTAR described below, and the second by algorithm BOUNDSTAR described in Section III-D2.

Complexes in stars $G(e|E_0)$ are maximally general, and therefore may describe objects in an overgeneralized way. The algorithm REDUSTAR generates a star and then maximally reduces the sparseness of each complex in it, while preserving the coverage of observed events. For example, complex α_3' in Fig. 1 is such a reduced complex obtained from complex α_3. The steps of the procedure are as follows.

1) *Elementary Stars, $G(e|e_i)$, $e_i \in E_0$, are Determined:* To generate an elementary star $G(e|e_i)$ of an event e against another event e_i, all variables that have different values in e than in e_i are identified. Suppose, with no loss of generality, that these variables are $x_1, x_2, \cdots, x_g$, and that $e_i = (r_1, r_2, \cdots, r_g, \cdots, r_n)$. The complexes of the star $G(e|e_i)$ are then $[x_j \neq r_j]$, $j = 1, 2, \cdots, g$, because these are the maximally general complexes which cover e and do not cover e_i. The number of complexes in an elementary star is at most n, and, because $e_i \neq e$, at least 1.

2) *The Complete Star $G(e|E_0)$ is Determined:* The star $G(e|E_0)$ is generated by first setting up the logical product $\Lambda \bar{G}(e|e_i)$, $e_i \in E_0$, where $\bar{G}(e|e_i)$ is the disjunction of complexes from the elementary star $G(e|e_i)$ [5]. Next, the multiplication of complexes is performed, using absorption laws, until a disjunction of irredundant complexes is obtained. This multiplication is carried out in steps, each step being a multiplication of a disjunction of complexes by a disjunction of selectors (the elements of consecutive elementary stars). The set of the complexes in the resulting disjunction is $G(e|E_0)$.

3) *Complexes in $G(e|E_0)$ are Reduced and Simplified:* The sparseness of each complex in the star is reduced as much as possible without "uncovering" any of the observed events. This is done by performing the REFUNION of all the observed events contained in each complex. The obtained complexes are then generalized and simplified.

C. NID Procedure

This procedure transforms a set of Nondisjoint complexes Into a set of Disjoint complexes (i.e., a disjoint clustering). If input complexes to NID are already disjoint, the procedure leaves them unchanged. The steps of the procedure are as follows.

1) *"Core" Complexes are Determined:* Observed events covered by more than one complex from the given set are placed on the *multiply-covered event list* (*m-list*). If the m-list is empty, then the complexes are only weakly intersecting, i.e., the intersection area contains only unobserved events. In this case, the procedure terminates with an indication that the combination of complexes is a *weakly intersecting clustering*. Otherwise, each complex is replaced by the REFUNION of the observed events contained in the complex that are not on the m-list (i.e., that are singly covered). The obtained refunions are called "core" complexes.

2) *A Best "Host" Complex is Determined for Each Event on the m-List:* An event is selected from the m-list and is "added" to each of the k core complexes by generalizing each complex to the extent necessary to cover the event. Such a generalization is performed by applying the REFUNION operator to the event and the complex. As a result, k modified complexes are obtained. By replacing one of the core complexes in the initial set with the corresponding modified complex, in k different ways, a collection of clusterings is obtained. These clusterings are evaluated according to the assumed clustering quality criterion (see next section). The complex in the best clustering that covers the given event from the m-list is considered to be the best "host" for this event. The best clustering is retained and the remaining ones are eliminated. By repeating the above operation for every event on the m-list, a set of k disjoint complexes is obtained whose union covers the same observed events as the original set of nondisjoint complexes.

If an event cannot be "added" to any complex without causing the result to intersect other complexes, then the event is placed on the *exceptions list.*

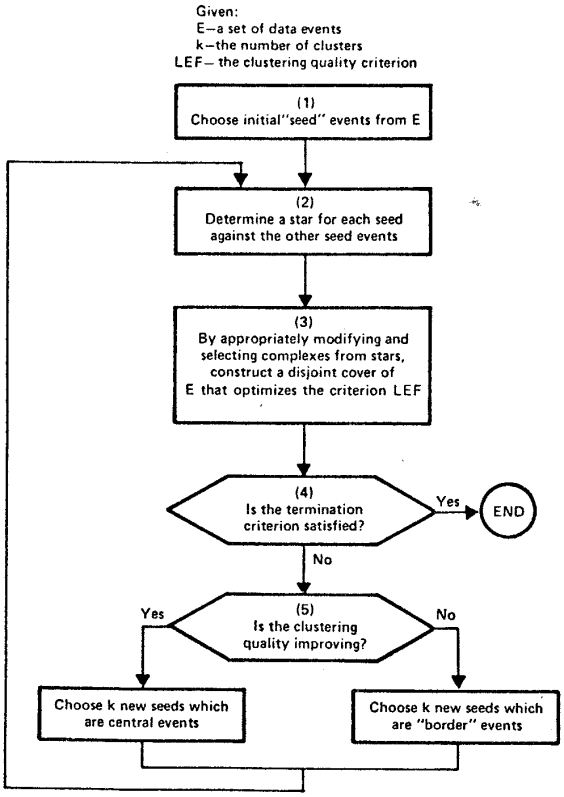


Fig. 2. The flowchart of the clustering module.

D. The Clustering Module

The basic algorithm underlying the implementation of the clustering module was introduced in [7] and its flowchart is presented in Fig. 2. Its purpose can be described as follows.

Given:

- a collection of events to be clustered
- the number of clusters desired (k)
- the criterion of clustering quality

Find: a disjoint clustering of the collection of events that optimizes the given criterion of clustering quality LEF.

We first describe a straightforward, exhaustive-search-based version of the algorithm, and then show how this version is modified to increase efficiency.

1) *The Exhaustive-Search Version of the Algorithm.*

The full-search version of the clustering algorithm is described here merely to provide insight into the difficulty of the problem and thus has only a theoretical value. It proceeds as follows.

1. Initial Seeds are Determined: From the given collection of events E, k events (the *initial seeds*) are selected. The seeds (After this first step, seeds are always selected according to certain rules; see step *5*.)

2. Stars are Constructed for Seeds: For each seed e_i, a reduced star $RG(e_i|E_0)$ is constructed by procedure REDUSTAR, where E_0 is the set of remaining seeds.

3. An Optimized Clustering (a Disjoint Cover of E) is Built by Selecting and Modifying Complexes from Stars: Every combination of complexes, created by selecting one complex from each star, is tested to see whether it contains intersecting complexes. If so, the complexes are modified by procedure NID to make them disjoint.

4. A Termination Criterion is Evaluated: If this is the first iteration, the obtained clustering is stored. In subsequent iterations the clustering is stored only if it scores better than previously stored clusterings according to the LEF (see Section II-E). The algorithm terminates when a specified number of iterations does not produce a better clustering (this number is defined by a *termination criterion*, described below).

5. New Seeds are Selected: New seeds are selected from sets of observed events contained in complexes of the generated clustering, one seed per complex. Two seed selection techniques are used. One technique selects "central" events, defined as events nearest the geometrical centers of the complexes. The other technique, stemming from the "adversity principle,"[3] selects "border" events, defined as events farthest from the centers. Ties for central or border events are broken in favor of events which have not been used recently as seeds. The technique of selecting central events is used repetitively in consecutive iterations as long as the clusterings improve. When the improvement ceases, border events are selected.

After selecting seeds, a new iteration of the agorithm begins from step *2*.

Along with a clustering, the algorithm generates k l-complexes describing individual clusters, and determines how these complexes score on the evaluation criteria in the LEF. The algorithm stops when the *termination criterion* is satisfied. The termination criterion is a pair of parameters (b, p), where b (the *base*) is a standard number of iterations the algorithm always performs, and p (the *probe*) is the number of additional iterations beyond b performed after each iteration which produced an improved cover. The general structure of the algorithm is based on the so-called dynamic clustering method [2], [3].

The most computationally costly part of this algorithm is the construction of an optimized clustering, given k seed events (step *3*). For an illustration, let us assume that $k = 2$ and that k "seeds," e_1 and e_2, have been selected from the collection E. In the first step, stars $G_1 = G(e_1|\text{remaining seeds})$ and $G_2 = G(e_2|\text{remaining seeds})$ are generated. Fig. 3 presents complexes of these stars as branches of a search tree. Branches from the root represent complexes of star G_1 that are $\alpha_{11}, \alpha_{12}, \cdots, \alpha_{1m_1}$, and branches at the second level (repeated m_1 times) represent complexes of star G_2 that are $\alpha_{21}, \alpha_{22}, \cdots, \alpha_{2m_2}$. Each combination of complexes, containing one complex from each star, corresponds to one path in the tree. Because any such combination may contain intersecting complexes, procedure NID is applied to each, and the result is a disjoint clustering. These clusterings are ordered according to the quality criterion LEF, and the best one is selected.

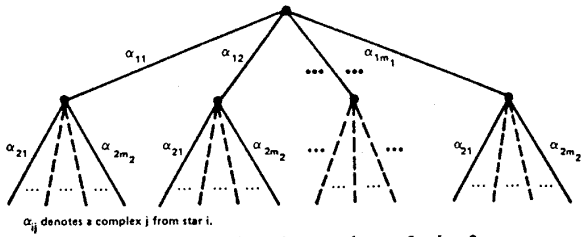


Fig. 3. The exhaustive search tree for $k = 2$.

2) *Path-Rank-Ordered (PRO) Search Procedure Used in CLUSTER/2.*

The above strategy for determining a clustering from seeds is very simple, but unfortunately too inefficient for solving any interesting practical problems. This is due to the fact that the stars may contain very many complexes. When there are n variables and k seeds, a star may contain up to n^{k-1} complexes (there are at most n complexes in any of the $k - 1$ elementary stars needed to compute the complete star). Thus, when $n = 30$ and $k = 3$, there could be up to $n^{k-1} = 900$ complexes, and the search tree could have up to 900-way branching at each node, and up to $900^3 = 729$ million leaves. Absorption laws (as defined in set theory) will usually eliminate many redundant complexes, but the star may still be too large. Artificial intelligence research on various heuristic search procedures offers various possibilities for reducing the search [14], [19]. To solve this problem, we have adopted some of the known ideas and also developed some new ones. The result is a search procedure called Path-Rank-Ordered (PRO) search that incorporates the following four techniques.

1. Bounding the Stars (Procedure BOUNDSTAR): The number of complexes in a star is bounded by a fixed integer m, which assures that the search tree has at most m-way branching. A bounded star contains not just m arbitrary complexes from the initial star, but the m "best" ones.

At each step of star generation (a multiplication of a set of complexes by the next elementary star, see STAR algorithm above), complexes are first reduced and then arranged in descending order according to the assumed clustering quality criterion LEF. Only the first m complexes are retained for the next multiplication step. This operation is also performed at the end of star generation, so that the final star has at most

[3]This principle states that if a border, "near hit" event truly belongs to the given cluster, then when selected as the seed it should produce a clustering that contains the same events as when a central event is used as a seed.

m complexes. The stars so obtained are called *bounded reduced stars* and denoted $G(e|E_0, m)$.

Some elementary criteria measure global properties of a clustering rather than properties of just a single complex (e.g., the disjointness). Consequently, when evaluating a complex descending from a node in the search tree that is not the root, the complex is evaluated in the context of complexes associated with the path from the root to this node.

By bounding the star we gain significantly in efficiency, but give up the assurance that the obtained clustering will be optimal. This is not a significant loss, however, because the clustering obtained at the end of each iteration contributes only the seeds to the next iteration, and thus its precise expression is not very important.

2. Generating Stars Dynamically: Because it is necessary to evaluate complexes in the context of previously selected complexes, bounding a star has to be done differently at each node of the search tree. CLUSTER/2 uses a "lazy" strategy, in which a star is generated only when it is needed to expand a node on the path being explored.

3. Searching in Order of Path Rank: As we mentioned above, complexes in a bounded star are arranged in descending order according to the LEF. In the search tree, the branch to the best complex is assigned the *branch index* 0, the branch to the next best complex is assigned the branch index 1, etc., up to the index $m - 1$. The *path index* of a path from the root to a leaf is the sum of the branch indexes along the path.

The paths from the root to a leaf represent potential clusterings and are investigated in the ascending order of their path index. Thus, the first path investigated is the one with path index 0, i.e., the path containing only the best complexes from each star. The next paths considered are those with a path index of one. There are k such paths.

As paths of increasing path index are generated and evaluated, a search termination criterion is applied. This criterion consists of two parameters, *search-base* and *search-probe*. A search-base number of paths is always expanded and evaluated. Then, a search-probe number of additional paths is considered. Each path is processed by NID, and if some complexes are transformed to make them disjoint, the clustering quality criterion is evaluated again. Whenever a new clustering is better than any previous clustering, it is saved and another search-probe number of additional paths is explored. If the above probing fails to find a better clustering, the search terminates.

4. Tapering the Search Tree: The bound of the stars m is decreased with the increase of the path index. The search tree is, therefore, more fully developed on the side containing the "higher quality" complexes.

Fig. 4 shows an example of a search tree generated by CLUSTER/2. The tree is a modification of the tree in Fig. 3, resulting from the application of the above efficiency-increasing techniques. In Fig. 4, the maximum bound m_{max} is set to 3. The root is expanded by constructing the star $G(\text{seed}_1|\text{other seeds}, 3)$, whose complexes are α_{11}, α_{12}, and α_{13} (listed in decreasing order of their "quality," as determined by the LEF). The branches representing these complexes are assigned branch indexes 0, 1, and 2, respectively. The node attached to branch 0 is expanded next. The star

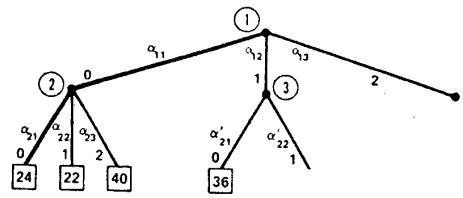


Fig. 4. The path-rank-ordered search tree for $k = 2$ used in CLUSTER/2.

$G(\text{seed}_2|\text{other seeds}, 3)$ is generated, creating complexes α_{21}, α_{22}, and α_{23}. Branches corresponding to these complexes are assigned branch indices 0, 1, and 2, respectively. The path 0-0 (having the lowest path index of 0, denoted by heavy lines in Fig. 4) is considered first. The associated clustering $\{\alpha_{11}, \alpha_{21}\}$ is processed by NID, and the result is saved as the best clustering so far. Next, path 0-1 is considered. The associated clustering $\{\alpha_{11}, \alpha_{22}\}$ is processed by NID and evaluated. If it is better than the previous clustering, it is saved. In order to explore the path 1-0 (the second path with path index 1), the star $G(\text{seed}_2|\text{other seeds}, 2)$ is generated. It contains complexes α'_{21} and α'_{22}. The clustering $\{\alpha_{12}, \alpha'_{21}\}$ associated with this path is evaluated. Assuming that the termination criterion has the parameters search-base = 2 and search-probe = 2, and that the evaluation scores are as shown in Fig. 4, the tree search terminates after investigating the fourth path 0-2 (since this path exhausts the probing without finding a better clustering). Path 0-1, with the evaluation score of 22, is the best clustering found.

E. The Hierarchy-Building Module

The hierarchy-building module uses the clustering module to determine a hierarchy of clusters. It performs two loops, one iterative and one recursive. The iterative loop repeats the clustering module for a sequence of values of k in order to determine the value for which the most desirable clustering is obtained. Such an approach is computationally acceptable because, in practical applications, most interesting hierarchies will have a relatively small number of branches (i.e., value of k) at each level.

The recursive loop applies the above iterative process at each node of the hierarchy. In the first step, the process is executed for the root, representing the initial event set E. Clusters of E and their conjunctive descriptions are determined. Consecutive steps repeat the same operation for the nodes representing clusters obtained during the previous step. The hierarchy continues to grow from the top down until the "continue-growth" criterion fails to be met. This criterion requires that the fit between the clusters and their descriptions at every level of the hierarchy must be better than at the previous level.

In order to determine the optimal value of k at each node, we must modify the clustering quality criterion so that it can be used to compare clusterings with different numbers of complexes. Such a criterion must reflect the dependency of the fit between the clustering and data on the value of k. As the

1. MP (Microprocessor)
Type: structured
Domain: 13 values
- 8080a
- 6502
- Z80
- 1802
- 6502C
- 6502A
- 68000
- 6800
- 6805
- 6809
- 8048
- Z8000
- HP (Hewlett Packard Co. proprietary)

2. RAM memory size
Type: linear
Domain: 4 values
- 16K bytes
- 32K
- 48K
- 64K

3. ROM memory size
Type: linear
Domain: 7 values
- 1K bytes
- 4K
- 8K
- 10K
- 11-16K
- 26K
- 80K

4. Display type
Type: structured
Domain: 4 values
- Terminal
- B/W_TV
- Color_TV
- Built-in

5. Keys on keyboard
Type: linear
Domain: 5 values
- 52 keys
- 53-56
- 57-63
- 64-73
- 92

Fig. 5. Variables used to describe microcomputers.

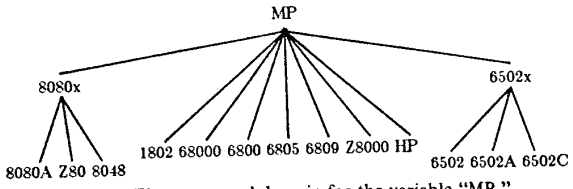


Fig. 6. The structured domain for the variable "MP."

number of clusters k increases, the fit (measured by the negative of sparseness) will likely increase, since smaller complexes will have smaller sparseness. On the other hand, increasing k increases the complexity of the clustering and therefore is undesirable. A simple criterion that takes into consideration the above tradeoff is to require the product

$$\text{total sparseness} \cdot k^{\beta}$$

to achieve minimum value, where β is an experimentally determined parameter balancing the relative effect of the sparseness and the number of clusters k on the solution.

IV. Comparison of Approaches

The described conjunctive conceptual clustering method will be compared with various numerical taxonomy methods using two exemplary problems, the first dealing with constructing a classification of popular microcomputers, and the second dealing with reconstructing a classification of selected soybean diseases.

A. Exemplary Problem I: Determining a Classification of Microcomputers

The problem is to develop a meaningful classification of popular microcomputers. Each microcomputer is described in terms of the variables shown in Fig. 5. "MP" and "Display type" are both structured variables with domains shown in Figs. 6 and 7, respectively. Descriptions of the microcomputers under consideration are presented in Fig. 8.

Two programs were applied to solve this problem:

1) NUMTAX, which implements 18 techniques of numerical taxonomy, and

2) CLUSTER/2, which implements conjunctive conceptual clustering.

The obtained results are described in the corresponding sections which follow.

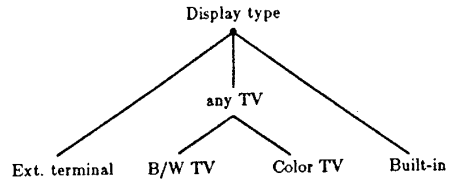


Fig. 7. The structured domain for the variable "display type."

1) *Results for Problem I using NUMTAX:* The principle of clustering applied by NUMTAX is that objects with high numerical similarity are placed in the same cluster while objects with low similarity are placed in different clusters. The numerical taxonomy program NUMTAX organizes the events into a hierarchy (a dendrogram) of clusters reflecting the similarities between consecutively larger groups of objects. In all, 18 such similarity-based techniques were used, spanning 3 data transformations, 3 numerical similarity measures, and 2 schemes for merging individual objects into clusters. Here are the selections that are available.

- Data Transformation Technique:
  a) none
  b) normalization to unit intervals
  c) standardization (values expressed in terms of standard deviations).
- Similarity Measure:
  a) product-moment correlation
  b) simple matching coefficients
  c) reciprocal Euclidean distance.
- Merging Criterion:
  a) unweighted average linkage
  b) weighted average linkage.

A particular technique is specified symbolically by three letters denoting specific choices of the data transformation, similarity measure, and the merging criterion. For example, ⟨*cba*⟩ denotes the combination using standardized data c), a similarity measure based on simple matching coefficients b), the unweighted average linkage merging criterion a).

The method generates a hierarchy in which the top level represents the complete collection of objects and the tips represent single objects. Dendrograms are constructed *bottom-up*, and to form top level clusters the entire dendrogram must always be generated. After this is done, the dendrogram is cut apart at the appropriate level to produce the desired number of clusters.

When the 18 dendrograms produced by NUMTAX are cut into two or three clusters, only 5 different partitionings are produced. A representative dendrogram produced by NUMTAX is shown in Fig. 9. In Fig. 9 the dashed lines indicate where the dendrogram is cut apart to form two and three clusters. Accompanying the dendrogram is a logical description of the clusters. These descriptions were produced by the inductive learning program AQ11 [10], which accepts as input a collection of groups (clusters) of objects and generates the simplest discriminant characterization of each group. The generated descriptions are in the form of a single *l*-complex or a logical disjunction of such complexes.

Fig. 9 shows the dendrogram produced by NUMTAX using

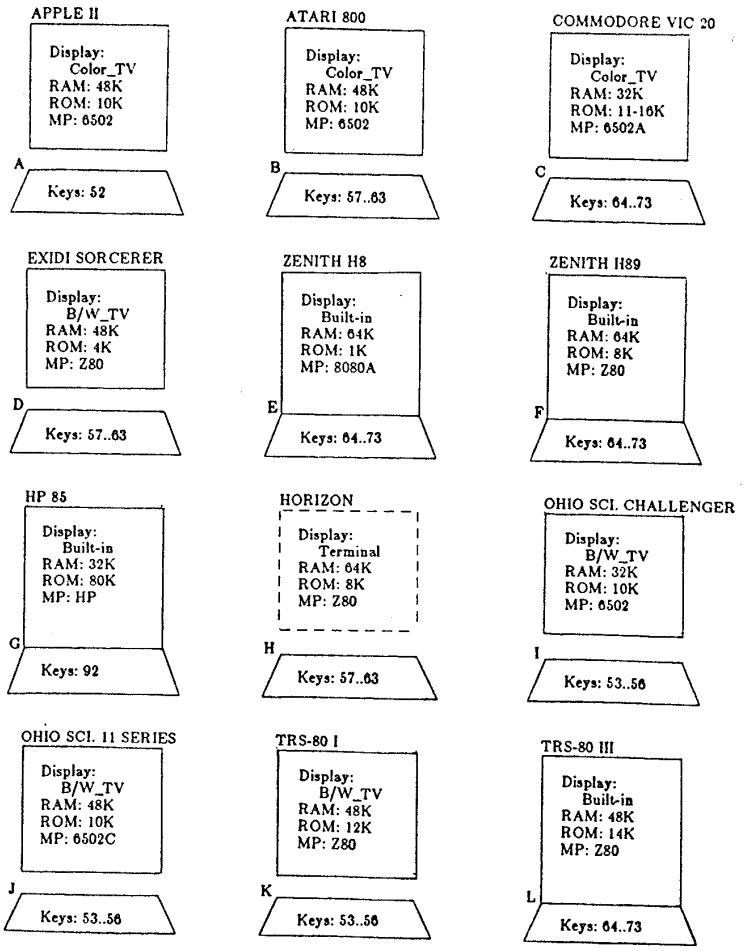


Fig. 8. Microcomputers.

the technique ⟨*aab*⟩ (i.e., using nontransformed data, product-moment correlation, and weighted average linkage). Very similar dendrograms were obtained using the techniques ⟨*abb*⟩, ⟨*aaa*⟩, and ⟨*aba*⟩ (the form of the dendrogram was identical; only some between-group similarity scores were slightly different). When the dendrogram was cut into two parts ($k = 2$), the clusters were $\{A, B, C, D, G, H, I, J, K, L\}$ and $\{E, F\}$. The descriptions produced by $AQ11$ for these clusters, as shown in Fig. 9, seem arbitrary. For example, the first cluster is described

$$[\text{RAM} = 16\text{K} \ldots 48\text{K}] \vee [\text{Keys} \leqslant 63].$$

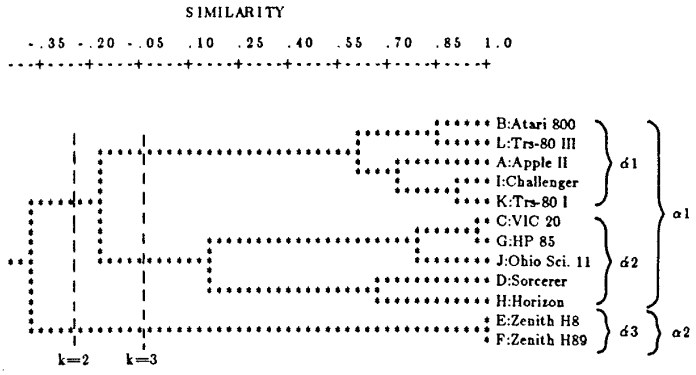
This description suggests that the cluster is composed of two kinds of computers, one that has [RAM = 16K . . 64K] and the other that has [Keys $\leqslant$ 63]. The presence of disjunction raises the question of the reason for placing these computers in the same cluster.

When the dendrogram is cut into three clusters ($k = 3$) the clusters are $\{B, L, A, I, K\}$, $\{C, G, J, D, H\}$, and $\{E, F\}$. The descriptions of these clusters involve entirely new variables (the type of microprocessor and the type of display) and appear again rather arbitrary and unrelated to the descriptions obtained in the two-cluster case.

2) Results for Problem I using CLUSTER/2: The principle of clustering applied by CLUSTER/2 is that objects are arranged into groups that are concisely circumscribed by conjunctive statements and optimized according to an assumed global criterion of clustering quality (LEF, Section III).

The program CLUSTER/2 was given the same data and was told to use three different evaluation criteria. The first LEF specified the criterion to maximize the disjointness between clusters with tolerance 0.3, and then to maximize the commonality with a tolerance of 0. The clustering obtained using this LEF is shown in Fig. 10. The second LEF specified the criterion to maximize the fit between the clustering and the events with tolerance 0.3, and then to maximize the simplicity of cluster descriptions with tolerance 0. The number of clusters to form was determined automatically with parameter $\beta = 4$ for the first level of the hierarchy and $\beta = 2$ for the second level. The clustering obtained using this LEF is shown in Fig. 11(a). The third LEF specified the same criterion as the second LEF but with parameter $\beta = 3$ for both hierarchy levels. The clustering obtained using this LEF is shown in Fig. 11(b).

The logical statements produced by CLUSTER/2 are always conjunctive, but may be quite long. They can be shortened by using the previously mentioned program $AQ11$, as was done to



(The dendrogram was cut as indicated to form two or three clusters)

Cluster descriptions generated by the program AQ11 are:

For the two-cluster solution (obtained by cutting the dendrogram at the dashed line marked by k=2)

$a1$: [RAM = 16K..48K] v [Keys ≤ 63]

$a2$: [RAM = 64K][Keys > 63]

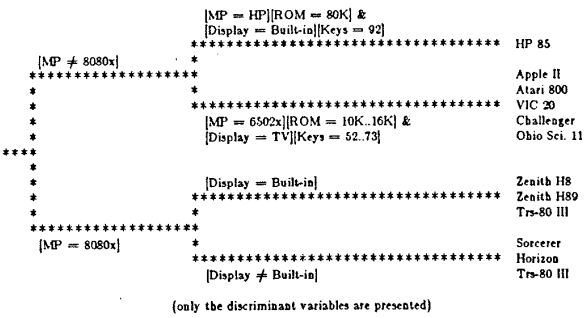
For the three-cluster solution (obtained by cutting the dendrogram at the dashed line marked by k=3)

$\acute{a}1$: [MP ≠ 6502C v 6502A v HP][ROM = 16K..80K]

$\acute{a}2$: [MP = 6502C v 6502A v HP] v [ROM = 1K..8K][Display ≠ Built-in]

$\acute{a}3$: [MP ≠ 6502C v 6502A v HP][ROM = 1K..8K][Display = Built-in]

Fig. 9. A dendrogram produced by NUMTAX in experiment I (clustering techniques ⟨*aab*⟩ ⟨*abb*⟩ ⟨*aaa*⟩ ⟨*aba*⟩).



(only the discriminant variables are presented)

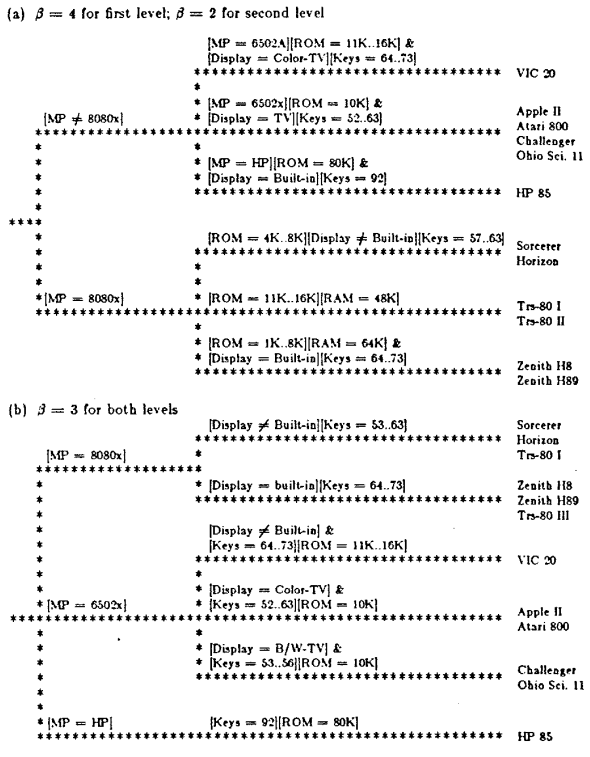
Fig. 10. A classification of microcomputers generated by CLUSTER/2 with an *a priori* number of clusters $k = 2$ for both levels. Clustering quality criterion LEF is "maximize both the disjointness and the commonality."

obtain the descriptions of clusters produced by NUMTAX. Another way to simplify the descriptions is to define two categories of variables:

- *common variables* which take the same value in each cluster, and
- *discriminant variables* which take different values for one or more clusters.

To simplify descriptions, all common variables are removed.

The first clustering (Fig. 10) is a two level hierarchy. This structure was obtained by imposing binary branching at each level of the tree. This arbitrary choice is useful when the hierarchical structure obtained will be compared to the dendrogram produced by conventional numerical taxonomy techniques. The other hierarchies produced by CLUSTER/2 that are presented were not constrained in this manner. The



(only the discriminant variables are presented)

Fig. 11. Classifications of microcomputers generated by CLUSTER/2 with an automatic determination of the number of clusters. Clustering quality criterion LEF is "maximize both the fit and the simplicity"; parameter β is as shown above.

program CLUSTER/2 can be used either to find a good conceptual clustering for a predetermined hierarchical form, or it can be used to find an optimized structure by automatically determining the number of clusters at each level.

At the first level in the hierarchy shown in Fig. 10, CLUSTER/2 split the microcomputers into two clusters according to the type of microprocessor. The choice of variables to use to conceptually differentiate the clusters is made by optimizing the clustering according to the LEF. In this case, one cluster is composed of microcomputers utilizing some form of 8080 microprocessor, as denoted by the generalized value 8080x which includes the 8080A, Z80, and 8048 microprocessors, while the other cluster contains the microcomputers utilizing a microprocessor that does not belong to the 8080 family.

At the second level of the hierarchy, CLUSTER/2 further partitioned the MP = 8080x cluster according to the microcomputer's display type. The other level-one cluster was partitioned according to four variables: microprocessor type, amount of ROM memory, display type, and the number of keys on the keyboard. The conceptual difference between the HP 85 and the other systems stands out clearly.

For the clustering shown in Fig. 11(a), CLUSTER/2 determined the optimized number of clusters to form at the first level of the hierarchy. The number $k = 2$ was determined by finding the k-clustering with the lowest value of sparseness $\cdot k^{\beta}$ for $\beta = 4$. This rather large value for β causes the program to demand a very great improvement in fit for an incremental increase in k. It seems reasonable, in practice, to use larger values of β at the top of the hierarchy and smaller values nearer the tips of the hierarchy. The second level was evaluated with $\beta = 2$. Given that each level-one cluster contains only 6 events, the limit on reasonable values of k at the second level is about 3 (i.e., two events per second level cluster, on the average). With $\beta = 2$, the optimized number of clusters for each part of the hierarchy at level 2 is $k = 3$.

In the second level of clusters on the MP $\neq$ 8080x branch of the hierarchy, the program determined that the ranges of numbers of keys 52..63, 64..73, and 92 fit well to the data. CLUSTER/2 uses the "closing the interval" generalization to form interval values of linearly ordered variables. A further step would be to name the interval values produced. In this example, the names "low range," "middle range," and "highest" could be applied to intervals to express the magnitude of the number of keys on the keyboards of the microcomputers.

The clustering hierarchy shown in Fig. 11(b) also contains two levels. At the first level, CLUSTER/2 split the microcomputers into $k = 3$ groups finding the value of k for which the criterion sparseness $\cdot k^{\beta}$ is minimal, with $\beta = 3$. (This same criterion selects $k = 2$ first-level clusters when $\beta = 4$, and $k = 6$ first-level clusters when $\beta = 2$.) The value $\beta = 3$ was also used to determine the optimized value of k for second-level clusterings under each first-level node. These values of k for the second-level clusterings are 2 and 3 for the two first-level clusters which contain more than one event. (The third first-level cluster contains only one event.)

The hierarchy shown in Fig. 11(b) reveals an underlying conceptual structure for the collection of microcomputers. Aided by background knowledge represented in the structure of the domain of the variable MP, CLUSTER/2 found that three microprocessor types: 8080x, 6502x, and HP were important for classifying these microcomputers. The categories "8080x" and "6502x" are generalized values of microprocessor type that cover the 8080 and 6502 families, respectively.

In level two, the MP = 8080x cluster is subdivided into two clusters that take different values for the variables Display (type) and (number of) Keys. Since either of these variables alone is sufficient to discriminate the clusters, this level-two clustering has a discrimination index of 2. In the three clusters stemming from the MP = 6502x cluster, there are no variables which alone can discriminate the clusters, however the pair of variables (Display, Keys) or the pair (Display, ROM) taken together is sufficient to provide complete discrimination. The hierarchical clustering clearly reveals the unique nature of the HP 85 microcomputer. Its values for the variables MP, Keys, and ROM are unique.

Of the 18 dendrograms generated by NUMTAX, only the four produced by techniques ⟨*baa*⟩, ⟨*bba*⟩, ⟨*bab*⟩, and ⟨*bbb*⟩ yielded a partitioning of the data into two clusters that was conceptually appealing. The numerical taxonomy techniques which produce such dendrograms cannot be predicted in advance. This is indicated by experiments involving several other sets of data. For example, the numerical taxonomy techniques which led to clusterings with simple conceptual descriptions failed to produce such simple descriptions for the "microorganisms" problem (Michalski, Stepp, and Diday [13]). One important result revealed by our study is that there is no one measure of numerical similarity that consistently leads to dendrograms having a simple conceptual interpretation. This is in contrast to conjunctive conceptual clustering which generates clusters with simple conceptual interpretations in all cases.

ENVIRONMENTAL VARIABLES		PLANT GLOBAL VARIABLES	
Time of occurrence	(7)	Severity	(3)
Plant stand	(3)	Seed treatment	(2)
Precipitation	(2)	Seed germination	(2)
Temperature	(4)	Plant height	(4)
Occurrence of hail	(3)		
Number of years crop repeated	(2)		
Damaged area	(3)		

PLANT LOCAL VARIABLES

Condition of leaves	(2)	Condition of stem	(3)
Leaf spots	(2)	Presence of lodging	(4)
Leaf spots margin	(2)	Stem cankers	(3)
Leaf spot size	(5)	Canker lesion color	(3)
Shotholing/shreading	(2)	Fruiting bodies on stem	(2)
Leaf malformation	(2)	External decay of stem	(3)
Leaf mildew growth	(2)	Mycelium on stem	(2)
Condition of seed	(3)	Internal discoloration	(3)
Seed mold growth	(4)	Sclerotia internal/external	(2)
Seed discoloration	(2)	Condition of fruit pods	(4)
Seed size	(2)	Fruit spots	(3)
Seed shriveling	(2)	Condition of roots	(3)

(integers in parentheses indicate the sizes of the domains)

Fig. 12. Multivalued variables used to describe cases of soybean disease.

B. Exemplary Problem II: Reconstructing a Classification of Soybean Diseases

The problem is to reconstruct a classification of selected soybean diseases. Given are 47 cases of soybean diseases each characterized by the 35 multivalued variables shown in Fig. 12.

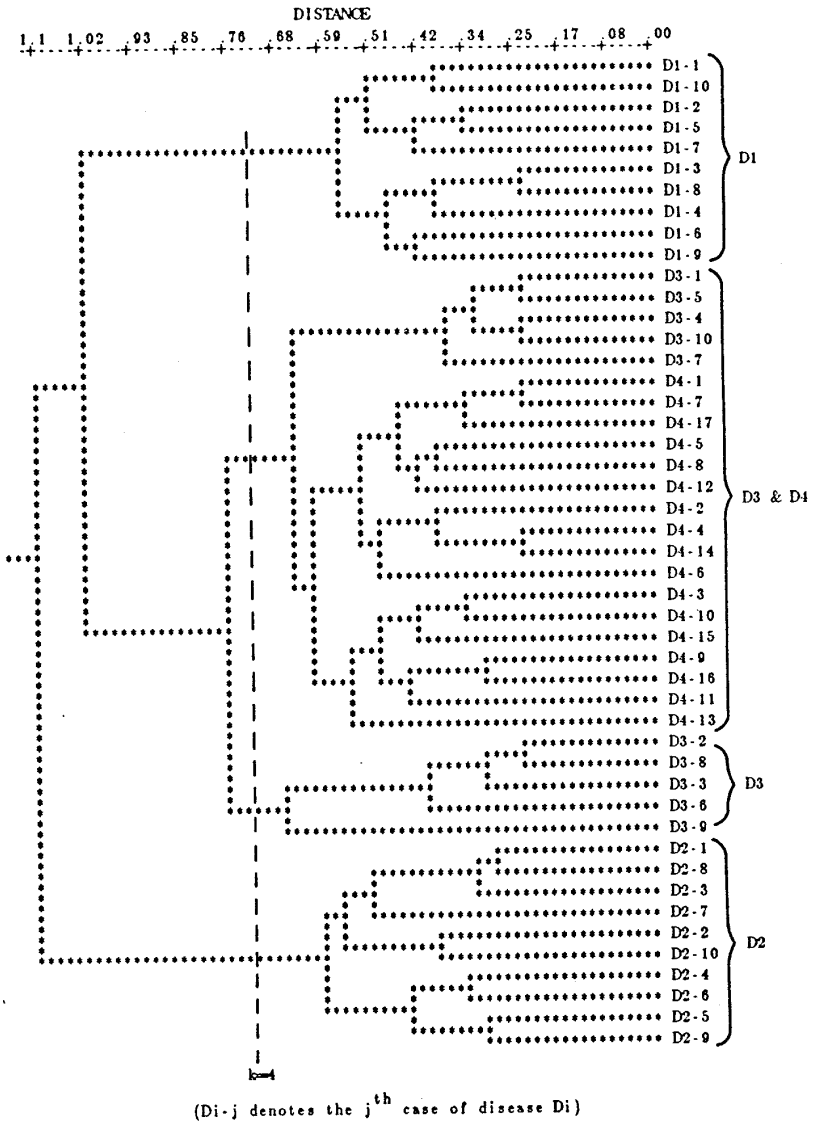


Fig. 13. Dendrogram of cases of soybean diseases $D1, D2, D3, D4$ using unweighted average linkage and Euclidean distance on nontransformed data.

These cases are drawn from 4 populations—each population representing one of the following soybean diseases:

$D1$—Diaporthe stem canker	$D3$—Rhizoctonia root rot
$D2$—Charcoal rot	$D4$—Phytophthora rot.

Ideally, a clustering method should partition these given cases into four groups corresponding to the diseases. To test this, we have applied program CLUSTER/2 and the 18 numerical taxonomy techniques, as in Example I, to cluster the given cases.

1) *Results for Problem II using NUMTAX:* Fig. 13 shows a typical dendrogram produced by the program NUMTAX (18 such dendrograms were obtained, one from each technique). As we see, this dendrogram separates correctly cases of diseases $D1$ and $D2$; however, cases of diseases $D3$ and $D4$ are somewhat intermixed. For $k = 4$ (Fig. 13) the cluster marked "$D3$ & $D4$" contains cases of both diseases $D3$ and $D4$. Only 4 of the 18 obtained dendrograms ($\langle cba \rangle$, $\langle cbb \rangle$, $\langle cca \rangle$, $\langle ccb \rangle$, i.e., those involving standardized data, product-moment correlation or simple matching scores, and average or weighted average linkage) precisely reconstructed the correct classification of the cases. The output from NUMTAX does not provide any description of the clusters formed.

2) *Results for Problem II using CLUSTER/2:* The program CLUSTER/2 was applied to this problem with "maximizing the fit" as the evaluation criterion (LEF). CLUSTER/2 partitioned the disease cases into four disease categories and described the clusters in terms of the characteristics (symptoms)

Variable	Range determined by CLUSTER/2	Range determined by plant pathologist
Precipitation	above normal	normal or above normal
Temperature	normal	normal or above normal
Stem cankers	above second node	above second node
Canker lesion color	brown or n.a.	brown
Fruiting bodies	present	present
Condition of fruit pods	normal	normal
Time of occurrence	July to October	August to September
No. yrs. crop repeated	several years	several years
Plant stand	normal	
External decay of stem	firm and dry	
Int. discolor. of stem	none	
Sclerotia int. or ext.	absent	
Condition of roots	normal	
Damaged area	scattered areas or low areas	
Severity	potentially severe or severs	
Leaf spots	absent	not present in expert descriptions
Shotholing/shreading	absent	
Leaf malformation	absent	
Leaf mildew growth	absent	
Condition of stem	abnormal	
Plant height	abnormal	
Condition of leaves	abnormal	
Mycelium on stem	absent	
Condition of seed	normal	
Seed treatment	none or fungicide	

Fig. 14. The description for one cluster (the disease diaporthe stem canker) obtained by CLUSTER/2 (variables having values in the left column) and as described by a plant pathologist (variables having values in the right column).

Variable	Cluster1 (Diaporthe stem canker)	Cluster2 (Charcoal rot)	Cluster3 (Rhizoctonia root rot)	Cluster4 (Phytophthora rot)
Precipitation	above normal	below normal	above normal	normal or above
Temperature	normal	normal or above	below normal	normal or below
Stem cankers	above 2nd node	absent	below soil line	below or slightly above soil
Canker lesion color	brown	tan	brown	dark br. or black
Fruiting bodies on stem	present	absent	absent	absent
Condition of fruit pods	normal	normal	few or none	irrelevant
Plant stand	normal	normal	irrelevant	less than normal
External decay of stem	firm and dry	absent	firm and dry	absent or firm and dry
Int. discolor. of stem	none	black	none	none
Sclerotia int. or ext.	absent	present	absent	absent
Condition of roots	normal	normal	normal or rotted	rotted
Damaged areas	scattered or low areas	whole fields, upland areas	low areas	whole fields, low areas

Fig. 15. Discriminant characteristics for clusters of soybean disease cases produced by CLUSTER/2.

of each disease, expressed in the form of a conjunctive statement. The produced disease categories corresponded exactly to actual soybean diseases, and the descriptions produced by CLUSTER/2 agreed well with the symptoms indicated by plant pathologists for these diseases.

Fig. 14 presents the complete *l*-complex for one cluster (one disease category). The middle column contains the values for the 25 variables CLUSTER/2 used to describe one cluster.

The right-hand column of Fig. 14 presents values of variables used by an expert plant pathologist to describe the same disease for diagnosis. The description of the disease determined by CLUSTER/2 contains all the symptoms of the disease specified by the plant pathologist (the values of "time of occurrence," "precipitation," and "canker lesion color" determined by CLUSTER/2 are supersets of the values mentioned by the plant pathologist). The description produced by CLUSTER/2 also involves many variables which the plant pathologist did not mention. Fig. 15 shows a table of the values of the discriminant variables for each cluster derived from the descriptions produced by CLUSTER/2.

Number of clusters	Sparseness ($\times 10^6$)	Parameter S β=2	β=3	β=4	Cpu time used (on Cyber 175)
2	15.00	60.0	120	240	10 sec
3	0.50	4.5	13.5	40.5	23
4	*0.03*	*0.5*	*1.9*	*7.7*	*21*
5	0.10	2.5	12.5	62.5	44
6	0.02	0.7	4.3	25.9	40

Fig. 16. A summary of evaluation criterion scores for soybean disease clusterings for $k = 2$ to 6.

The measure of the total sparseness of the solution can be used to judge the best number of clusters to form. Data from the clustering of soybean disease cases for $k = 2$ through $k = 6$ are summarized in Fig. 16. As k increases, the sparseness always decreases because data events are partitioned into smaller complexes which fit the data better. On the other hand, increasing k is undesirable as it raises the complexity of the clustering. A measure that reflects this tradeoff is S = sparseness $\cdot k^{\beta}$, where β is a parameter which balances the influence of k versus the sparseness. Fig. 16 shows values of parameter S for $\beta = 2$ through $\beta = 4$. In our experiment there was a strong correlation between the cpu time used and the parameter S. This fact may indicate that the algorithm operates more efficiently when the number of clusters formed agrees with some "natural" organization of the data.

The clusterings obtained from CLUSTER/2 depend on components of the assumed criterion of clustering quality. One can, therefore, possibly argue that this is equivalent to obtaining different clusterings by using different methods of measuring object similarity. There are, however, two important differences in this regard between the conjunctive conceptual clustering method and traditional methods.

The first difference is that conceptual clustering provides the data analyst with a simple conceptual description of the generated clusters. Consequently, it is easy to experiment with the choice of the clustering quality criterion, and to determine the clusterings most suitable to the problem under consideration. The second difference is that the clustering quality criterion, unlike a measure of similarity, has a clear and simple relationship to the clusters to be generated, as its components represent elementary criteria such as the "fit" of the clusters to the data, the "simplicity" of cluster descriptions, etc.

V. Summary

A method of conjunctive conceptual clustering was analyzed and compared to a number of clustering techniques used in numerical taxonomy. The major difference between this method and numerical taxonomy methods is that it performs clustering not on the basis of some mathematical measure of object similarity, but on the basis of "concept membership." Specifically, clusters are groups of objects which are characterized by simple, "well fitted" conjunctive descriptions. The collection of cluster descriptions optimizes a predefined global clustering quality criterion. Experiments performed so far have shown that the method produces clusters that tend to match solutions most satisfactory for people. Similar experiments with numerical taxonomy methods resulted in clusters

that were less satisfactory in this regard. The numerical taxonomy methods produced clusters that in the majority of cases seemed to be arbitrary and rather inadequate from the viewpoint of human interpretation. This can be explained by noting that the numerical taxonomy program NUMTAX is not equipped with any knowledge of human conjunctive concepts (or any other concepts) and, therefore, cannot knowingly produce clusters corresponding to such concepts.

From the viewpoint of traditional clustering methods, conceptual can be interpreted as an approach that also uses a certain measure of object similarity, but of a quite different kind. This new kind of similarity measure takes into consideration not only the distance between objects (as in conventional clustering methods), but also their relationship to other objects and, most importantly, their relationship to some predetermined concepts (here, conjunctive descriptions).

The price for using such a "concept-dependent" similarity measure is a significantly greater computational complexity of the method, and consequently, the run time of the clustering program. For example, each dendrogram produced by NUMTAX (implemented in Fortran) for example I required about 60 ms of processor time on a Cyber 175, while clusterings produced by CLUSTER/2 (implemented in Pascal) for the same example required 4-40 s of processor time. (The above comparison is not totally appropriate because NUMTAX produces only clusters, while CLUSTER/2 produced both clusters and their descriptions.) The greater computational complexity is not necessarily a significant disadvantage of the method. If the clusterings obtained are indeed useful and practical, then the computational cost is of little relevance, especially now when the prices of computer technology are declining. Experience shows that researchers using presently available clustering techniques are most concerned not with the amount of computational time expended, but with the difficulty of interpreting the results of the analysis.

Another important characteristic of the method (and a limitation or advantage depending on the problem at hand) is that it is specifically oriented toward clustering problems using nominal or ordinal variables. It should be noted, however, that the method can also handle other types of variables, if they are properly quantized.

The major current limitation of the presented method of conjunctive clustering (as well as all traditional methods) is that it clusters objects using only the variables that were defined in the input data. This limitation could be overcome by a process of "constructive induction" that incorporates into cluster descriptions new variables, derived as certain functions of the initial ones [6], [8].

In conclusion, the presented method of conjunctive conceptual clustering adds a new dimension to research in cluster analysis, and seems to have the potential to be a useful new tool for researchers analyzing data.

ACKNOWLEDGMENT

The authors wish to thank Prof. R. Selander, Department of Genetics, University of Illinois, for providing the numerical taxonomy program NUMTAX used in the comparative analysis of clustering methods.

REFERENCES

[1] M. R. Anderberg, *Cluster Analysis for Applications.* New York: Academic, 1973.

[2] E. Diday and J. C. Simon, *Clustering Analysis: Communication and Cybernetics*, vol. 10. New York: Springer-Verlag, 1976, pp. 47-92.

[3] E. Diday, G. Govaert, Y. Lechevallier, and J. Sidi, "Clustering in pattern recognition," in *Proc. 5th Int. Conf. Pattern Recognition*, Miami Beach, FL, Dec. 1-4, 1980, pp. 424-429.

[4] R. S. Michalski, "Variable-valued logic: System VL_1," in *Proc. 1974 Int. Symp. Multiple-Valued Logic*, West Virgina Univ., Morgantown, WV, May 29-31, 1974, pp. 323-346.

[5] —, "Variable-valued logic and its applications to pattern recognition and machine learning," in *Multiple-Valued Logic and Computer Science*, D. Rine, Ed. Amsterdam: North-Holland, 1975, pp. 506-534.

[6] —, "Pattern recognition as rule-guided inductive inference," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-2, no. 4, pp. 349-361, 1980.

[7] —, "Knowledge acquisition through conceptual clustering: A theoretical framework and an algorithm for partitioning data into conjunctive concepts," Special Issue on Knowledge Acquisition and Induction, *Int. J. Policy Anal. Inform. Syst.*, vol. 4, no. 3, pp. 219-244, 1980.

[8] —, "A theory and methodology of inductive learning," in *Machine Learning: An Artificial Intelligence Approach*, R. S. Michalski, J. Carbonell, T. Mitchell, Eds. Palo Alto, CA: Tioga, 1983.

[9] R. S. Michalski and R. L. Chilausky, "Learning by being told and learning from examples: An experimental comparison of the two methods of knowledge acquisition in the context of developing an expert system for soybean disease diagnosis," *Int. J. Policy Anal. Inform. Syst.*, vol. 4, no. 2, pp. 125-161, 1980.

[10] R. S. Michalski and J. B. Larson, "Selection of most representative training examples and incremental generation of VL_1 hypothesis: The underlying methodology and the description of programs ESEL and AQ11," Dep. Comput. Sci., Univ. Illinois, Urbana, Rep. 867, 1978.

[11] R. S. Michalski and R. Stepp, "Revealing conceptual structure in data by inductive inference," in *Machine Intelligence*, vol. 10, J. Hayes, D. Michie, and Y.-H. Pao, Eds. Chichester: Ellis Horwood, New York: Halsted, 1981.

[12] —, "An application of AI techniques to structuring objects into an optimal conceptual hierarchy," in *Proc. 7th Int. Joint Conf. Artificial Intell.*, Vancouver, Canada, Aug. 24-28, 1981, pp. 460-465.

[13] R. S. Michalski, R. E. Stepp, and E. Diday, "A recent advance in data analysis: Clustering objects into classes characterized by conjunctive concepts" (Invited chapter), in *Progress in Pattern Recognition*, vol. 1, L. Kanal and A. Rosenfeld, Eds., 1981, pp. 33-55.

[14] N. T. Nilsson, *Principles of Artificial Intelligence.* Palo Alto, CA: Tioga, 1980.

[15] R. R. Sokal and P. H. Sneath, *Principles of Numerical Taxonomy.* San Francisco, CA: W. H. Freeman, 1963.

[16] R. Stepp, "Learning without negative examples via variable-valued logic characterizations: The uniclass inductive program AQ7UNI," Dep. Comput. Sci., Univ. Illinois, Urbana, Rep. 982, July 1979.

[17] —, "A description and user's guide for CLUSTER/PAF—A program for conjunctive conceptual clustering," Dep. Comput. Sci., Univ. Illinois, Urbana, Rep. UIUCDCS-R-82-1084, 1982.

[18] S. Watanabe, *Knowing and Guessing: A Quantitative Study of Inference and Information.* New York: Wiley, 1969.

[19] P. H. Winston, *Artificial Intelligence.* Reading, MA: Addison-Wesley, 1977.

[20] N. G. Zagoruiko and G. S. Lbov, "Algorithms of pattern recognition in a package of applied programs," in *Proc. 4th Int. Conf. Pattern Recognition*, Kyoto, Japan, 1978, p. 1100.

Ryszard S. Michalski was born in Kalush, Poland, on May 7, 1937. He studied at the Cracow and Warsaw Technical Universities, and received the M.Sc. degree from the Leningrad Polytechnical Institute in 1961 and the Ph.D. degree from the University of Silesia in 1969.

From 1962 to 1970 he was associated with the Institute of Auto-

matic Control of the Polish Academy of Sciences in Warsaw, where he worked first as a research scientist and subsequently as the Leader of the Pattern Recognition Group. He joined the University of Illinois at Urbana-Champaign in 1970, where he is now a Professor of Computer Science and Medical Information Science. His research interests include inductive learning, expert systems, databases, plausible reasoning, conceptual data analysis, and applications of artificial intelligence to medicine and agriculture. He has published some 70 research and technical papers on these and related subjects, and recently coedited a book on machine learning. In 1982 he was appointed as an Associate at the Center of Advanced Study at the University of Illinois.

Robert E. Stepp was born in Lincoln, NE, on April 15, 1948. He received the B.A. and M.Sc. degrees from the University of Nebraska-Lincoln, in 1970 and 1972, respectively.

He is expecting the Ph.D. degree in 1983 from the University of Illinois, Urbana-Champaign. From 1972 to 1976 he was Manager of Systems Programming at the University of Nebraska-Lincoln. From 1981 to 1982 he was Instructor of Computer Science at the University of Illinois. He is currently a Visiting Research Associate at the Department of Computer Science, University of Illinois. His research interests include inductive learning, expert systems, conceptual data analysis, and computerized aids for the handicapped.