KNOWLEDGE ACQUISITION THROUGH CONCEPTUAL CLUSTERING: A THEORETICAL FRAMEWORK AND AN ALGORITHM FOR PARTITIONING DATA INTO CONJUNCTIVE CONCEPTS

by

R. S. Michalski

Journal of Policy Analysis and Information Systems, Vol. 4, No. 3, September 1980. and Report No. UIUCDCS-R-80-1026, Department of Computer Science, University of Illinois, Urbana.

International Journal of Policy Analysis and Information Systems, Vol. 4, No. 3, 1980

Knowledge Acquisition Through Conceptual Clustering: A Theoretical Framework and an Algorithm for Partitioning Data into Conjunctive Concepts[1]

Ryszard S. Michalski[2]

Received April 29, 1980; revised May 20, 1980

The conventional methods of cluster analysis partition given entities into clusters of "similar" entities, using a similarity function which takes into consideration only the information about the entities themselves. Therefore, clusters obtained this way do not usually have any simple conceptual interpretation. The paper presents an approach to clustering (called conceptual clustering), in which entities are assembled into a single cluster not because of their pairwise similarity, but because together they represent a concept from a predefined set of concepts. In the presented theory and algorithm PAF, the concepts characterizing clusters are single conjunctive statements involving relations on variables which describe the entities. Thus, the algorithm not only clusters entities, but also provides descriptions of the obtained clusters. The algorithm is iterative and its general structure is based on the dynamic clustering method. In one of the testing examples, the implemented algorithm was able to re-discover the correct classification of an unordered collection of cases of four different soybean diseases.

KEY WORDS: Cluster analysis; data analysis; learning without teacher; knowledge acquisition; numerical taxonomy; pattern recognition; inductive inference; classification theory; conceptual clustering.

Then he took the seven loaves
and the fish, and when he had given
thanks, he broke them and gave them
to the disciples, and they in turn
to the people.
Matthew 15:36

[1]Partial support of this research was provided by the National Science Foundation under Grant MCS-79-06614, the Université Paris IX (Dauphine), and the IRIA Research Institute in France.

[2]Department of Computer Science, University of Illinois, Urbana, Illinois 61801.

1. INTRODUCTION

Clustering is the intelligent partitioning of a collection of entities. Specifically, it is the process of dividing entities (objects, observations, measurements, data, etc.) into categories that are meaningful or useful for some purpose. It is one of the fundamental operations people use to simplify descriptions of their environment, and by that, to improve the efficiency of their decision making. Appropriate clustering reveals the underlying structure of the given set of objects, and hence clustering can be viewed as a form of knowledge acquisition.

Clustering problems pervade many fields, particularly experimental sciences such as biology, chemistry, geology, and medicine. Intelligent partitioning of objects can also be an important capability of autonomous or semiautonomous robots designed for exploration of special environments (e.g., the bottom of an ocean or the surface of a planet). Consequently, understanding the nature of clustering is not only of scientific interest but also of significant practical importance.

A conventional view of clustering is that it is a process of partitioning objects into groups such that the degree of similarity (or "natural association") is high among objects of the same group, and low among the objects of different groups. The notion of the degree of similarity between objects is therefore fundamental to this viewpoint. A great variety of different similarity measures have been developed and used in various clustering techniques. Frequently a reciprocal of a distance measure is used as a similarity function. The distance measure for such purposes, however, does not have to satisfy all the postulates of a distance function (specifically, the triangle inequality). A comprehensive review of various distance and similarity measures is provided by Diday and Simon[1] and Anderberg.[2] Becker[3] describes a fuzzy similarity measure based on the theory of fuzzy sets.

To determine the similarity of objects, a measure of similarity is applied to symbolic descriptions of objects (data points). Such descriptions are typically vectors, whose components represent scores on selected qualitative or quantitative variables used to describe objects. The underlying assumption is that if the similarity function has high value for the given descriptions, then the objects represented by the descriptions are similar. The similarity relationship between any two objects in the population to be clustered is thus reduced to a single number—the value of the similarity function applied to symbolic descriptions of objects.

Conventional measures of distance are "context-free," i.e., the distance between any two data points A and B is a function of these points only, and does not depend on the relationship of these points to other data points:

$$\text{Similarity}(A, B) = f(A, B) \tag{1}$$

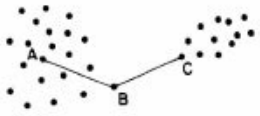

Fig. 1. An illustration of the context-free distance.

For example, for any conventional distance measure, the distance between points A and B in Fig. 1 is the same as between B and C although A and C have a different relationship to the remaining data points.

Recently some authors have been introducing "context-sensitive" measures of similarity:

$$\text{Similarity}(A, B) = f(A, B, E) \tag{2}$$

where the similarity between A and B depends not only on A and B, but also on the relationship of A and B to other data points, represented in (2) by E.

For example, Gowda and Krishna[4] defined the so-called "mutual neighborhood" distance measure. If point A is the nth closest point to B and B is the mth closest point to A, then the mutual neighborhood distance between A and B is $n + m$. These authors have demonstrated that a method using such a distance measure can solve some clustering problems which methods based on the "context-free" distance cannot.

Both previous clustering approaches cluster data points only on the basis of knowledge of the individual data points. Therefore such methods are fundamentally unable to capture the "Gestalt property" of objects, i.e., a property which is characteristic to certain configurations of points considered as a whole, and not as a collection of independent points. In order to detect such properties, the system must know not only the data points, but also certain "concepts." To illustrate this point, let us consider a problem of clustering data points in Fig. 2.

A person considering the problem in Fig. 2 would typically describe it as "a circle on top of a rectangle." Thus, the points A and B, although being very close, are placed in separate clusters. Here, human solution involves partitioning the data points into groups not on the basis of pairwise distance between points, but on the basis of "concept membership." That means that the points are placed in the same cluster if together they represent the same concept. In our example, the concepts are a circle and a rectangle.

The approach to clustering which clusters objects into groups representing *a priori* defined conceptual entities is called "conceptual clus-

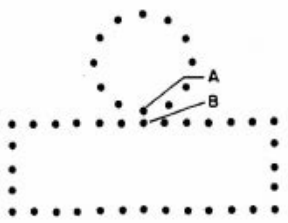

Fig. 2. Clustering into concepts: a circle on top of a rectangle.

tering." A link between conceptual clustering and distance-based clustering methods can be established by stating that in conceptual clustering the similarity between the data points is a function of these points, context E, and a set of predefined concepts C:

$$\text{Similarity}(A, B) = f(A, B, E, C) \tag{3}$$

The approach has been introduced by Michalski.[5] It evolved from earlier work by the author and his collaborators on the problem of generating "uniclass covers." Such covers are disjunctive descriptions of a class of objects learned from only positive examples of the class. Stepp[6] describes a computer program and various experimental results on determining uniclass covers. His work is concerned with what can be called "free"[3] conceptual clustering.

The idea that the similarity measures of the type (1) or (2) (the "concept-free" measures) may be inadequate for some clustering problems is not new. In the past, several authors noticed this problem it and proposed various solutions. For example, Watanabe[7,8] proposed the concept of "cohesion" to measure the "degree of clusterness" of points, which utilizes the entropy measure. Using this concept he was able to resolve the "three girls in the dormitory" paradox, which cannot be solved by "concept-free" methods. Other measures of "cohesiveness" of objects were proposed on the basis of graph-theoretic considerations, e.g., by Matula,[9] Auguston and Minker,[10] Zahn,[11] and Cheng.[12]

This paper presents a theoretical basis and an algorithm for conceptual clustering, where conceptual entities are conjuctive statements in variable-valued logic calculus VL_1 [13] (which is a typed many valued logic extension of propositional calculus). These statements, called VL_1 complexes, are logical products of relational statements involving discrete variables of an arbitrary number of values (Definitions 2 and 3 in Sec. 2). Complexes have a simple linguistic interpretation and are able to express consisely a large class of relationships among discrete variables. The algorithm combines the methodology of optimization of variable-valued logic expressions[14] with the dynamic clustering method.[1] Its theoretical foundation is a special property of complexes formulated as the Sufficiency Principle (Sec. 3).

2. COMPLEXES AS CONCEPTUAL ENTITIES FOR CLUSTERING: BASIC DEFINITIONS

Let $x_1, x_2, \ldots, x_n$ denote discrete variables which are selected to describe objects in the population to be clustered. For each variable a *value*

[3]In "free" clustering the number of clusters is not predefined, as opposed to "constraint" clustering where the number of clusters is assumed *a priori*.

set or *domain* is defined, which contains all possible values this variable can take for any object in the population. We shall assume that the value sets of variables x_i, $i = 1, 2, \ldots, n$, are finite, and therefore can be represented as

$$D_i = \{0, 1, \ldots, d_i\}, \qquad i = 1, 2, \ldots, n \tag{4}$$

In general, the value sets may differ not only with respect to their size, but also with respect to the structure relating their elements (reflecting the scale of measurement). In this paper we will restrict ourselves only to the case of nominal or linear variables, i.e., variables with unordered or linearly ordered domains, respectively. A sequence of values of variables x_1, $x_2, \ldots, x_n$, is called an *event*:

$$e = (r_1, r_2, \ldots, r_n) \tag{5}$$

where $r_i \in D_i$, $i = 1, 2, \ldots, n$.

The set of all possible events, **E**, is called the *event space*:

$$\mathbf{E} = \{e_i\}_{i=1}^{d} \tag{6}$$

where $d = d_1 \cdot d_2 \cdot \cdots \cdot d_n$ (the size of the event set) and $d_i = d_i + 1$.

Definition 1. Given two events e_1, e_2 in **E**, the *syntactic distance* $\delta(e_1, e_2)$ between e_1 and e_2 is defined as the number of variables which have different values in e_1 and e_2.

Definition 2. A relational expression

$$[x_i \# R_i] \tag{7}$$

where R_i, called the *reference set*, is one or more elements from the domain D_i and $\#$ stands for one of the relational operators $=$, $\neq$, $\geq$, or $\leq$, is called a VL_1 *selector*[4] or, briefly, a *selector*.

Here are a few examples of a selector, in which variables and their values are represented by linguistic terms:

[height = tall]

[color = blue, red] (read: color is blue or red)

[length $\geq$ 2]

[size $\neq$ medium]

[weight = 2 . .5]

The operator . . in the last selector denotes the range of values from 2 to 5, inclusively. It is used when the domain of the variable is a linearly ordered

[4] VL_1 stands for variable-valued logic system one,[13] which uses such selectors.

set. A selector $[x_i \# R_i]$ is said to be *satisfied* by an event $e = (r_1, r_2, \ldots, r_n)$ if r_i, i.e., the value of x_i in e is in relation $\#$ with any element of R_i.

Definition 3. A logical product of selectors is called a VL_1 *term*:

$$\bigwedge_{i \in I} [x_i \# R_i] \tag{8}$$

where $I \subseteq \{1, 2, \ldots, n\}$ and $R_i \subseteq D_i$. A set of events which satisfy a VL_1 term is called a VL_1 *complex* or, briefly, a *complex*.

Thus a VL_1 term is a formal representation of a complex. Since these two notions have a one-to-one correspondence, we will use them interchangeably, unless it leads to a confusion. Therefore, if a set-theoretic notation is applied to a term, it means that the operation is applied to the corresponding complex (i.e., a set of events satisfying the term). A complex (VL_1 term) α is said to *cover* an event e, if the values of variables in e satisfy the relational statements (selectors) in the complex (term).

For example, event $e = (2, 7, 0, 1, 5, 4, 6)$ satisfies the complex $[x_1 = 2, 3][x_3 \leq 3][x_5 = 3 \,..\, 8]$.

Let E be a set of events in $\mathbf{E}$, which are data points to be clustered. The events in E are called *data events* (or *observed events*) and events in $\mathbf{E} \backslash E$ (i.e., events in $\mathbf{E}$ which are not data events) are called *empty events* (or *unobserved events*).

Let α be a complex which covers some data events and some empty events.

Definition 4. The number of empty events covered by α is called the *sparseness* of α and is denoted by $s(\alpha)$.

Let $p(\alpha)$ denote the number of data events covered by α, and let $t(\alpha)$ denote the total number of events covered by α. We have then $t(\alpha) = p(\alpha) + s(\alpha)$. The total number of events satisfying the complex $\alpha = \bigwedge_{i \in I} [x_i \# R_i]$ is

$$t(\alpha) = \prod_{i \in I} c(R_i) \cdot \prod_{i \notin I} d_i \tag{9}$$

where $I \subseteq \{1, 2, \ldots, n\}$, $c(R_i)$ is the cardinality of R_i, and d_i is the cardinality of the value set of variable x_i.

Definition 5. The *degree of generality* $g(\alpha)$ of complex α is defined as follows:

$$g(\alpha) = \log \frac{t(\alpha)}{p(\alpha)} = \log \left(1 + \frac{s(\alpha)}{p(\alpha)}\right) \tag{10}$$

The value $t(\alpha)/p(\alpha)$ specifies how many events are in the complex per one data event. Thus, the degree of generality $g(\alpha)$ expresses the uncertainty of the location of the data points in the complex. The greater the degree of generality of a complex, the greater is the uncertainty. If $g = 0$, then all the events in the complex are data events. If $y = 1$, there are as many unobserved events in the complex as there are data events. We can see from (10) that for a fixed $p(\alpha)$, the degree of generality is a monotonic function of sparseness.

Let L be a set of complexes (or events), and R_i be the set of all the distinct values which variable x_i takes in these complexes (or events).

Definition 6. The operation which transforms L into the complex $\bigwedge_{i=1}^{n} [x_i = R_i]$ is called *reference union* or *refunion*. The resulting complex is called the *minimal covering complex* or *mc-complex* for L and denoted $RU(L)$ (refunion).

If any $R_i = D_i$, then the corresponding selector is removed from the complex. The refunion is thus a transformation which transforms a set of complexes into the minimal covering complex.

Theorem 1. The mc-complex of an event set has the minimum sparseness among all complexes covering this set.

Proof. Let α be the mc-complex for an event set E:

$$\alpha = RU(E) = \bigwedge_{i=1}^{n} [x_i = R_i] \tag{11}$$

where $R_i \subseteq D_i$ (the domain of x_i). Suppose that $\gamma = \bigwedge_{i=1}^{n} [x_i = P_i]$ is a complex which covers E and has a smaller sparseness than α. If this is true, then there must exist P_i such that $P_i \subset R_i$. But R_i, according to Definition 6, contains all values that x_i takes in events in E. Therefore, if $P_i \subset R_i$, then complex α could not possibly cover all events in E, which is a contradiction. ∎

Let E be data events which are covered by a complex α.

Definition 7. The set E is called the *core* of α, and the complex $\alpha^* = RU(E)$ is called the *trimmed* α.

From Theorem 1 we have $\alpha^* \subseteq \alpha$.

Theorem 2. If E_1 and E_2 are two disjoint event sets then

$$s(RU(E_1)) + s(RU(E_2)) \leq s(RU(E_1 \cup E_2)) \tag{12}$$

Proof. According to Theorem 1, $RU(E_1)$ and $RU(E_2)$ have the smallest possible sparseness among all complexes covering E_1 and E_2, respectively. Since E_1 and E_2 are disjoint, then (12) must hold. ∎

The property expressed by Theorem 2 has an analogy in statistical clustering, where with the increasing number of clusters the "fit" between each cluster and the probability distribution "fitted" to the cluster also increases.

Theorem 3. Let α_1 and α_2 be two intersecting complexes, whose union covers an event set E. Let E_1 (E_2) denote the set of events in α_1 (α_2) which are covered only by this complex (the *relative core* of the complex). Let α_1' and α_2' be any two disjoint complexes covering the same event set E. If $RU(E_1)$ and $RU(E_2)$ are disjoint complexes, then:

$$s(RU(E_1))+s(RU(E_2)) \leq s(\alpha_1')+s(\alpha_2') \tag{13}$$

Proof. The theorem is an immediate consequence of Theorem 2 and the premise that α_1' and α_2' are disjoint complexes. ∎

We will next introduce two basic concepts for the conceptual clustering algorithm presented in Sec. 6. They are the *star* of an event against an event set and a *cover* of an event set against another event set.

Let F be a proper subset of the event space $\mathbf{E}$, and e an event outside of F, i.e., $e \notin F$.

Definition 8. The *star* $G(e|F)$ of e against F is the set of all maximal under inclusion complexes covering the event e and not covering any event in F. (A complex α is *maximal under inclusion with respect to property P* if there does not exist a complex α^* with property P such that $\alpha \subset \alpha^*$.)

Let E_1 and E_2 be two disjoint event sets, $E_1 \cap E_2 = \emptyset$.

Definition 9. A cover $COV(E_1|E_2)$ *of* E_1 *against* E_2 is any set of complexes, $\{\alpha_j\}_{j \in J}$, such that for each event $e \in E_1$ there is a complex α_j, $j \in J$, covering it, and none of the complexes α_j cover any event in E_2. Thus we have

$$E_1 \subseteq \bigcup_{j \in J} \alpha_j \subseteq \mathbf{E} \setminus E_2 \tag{14}$$

A cover in which all complexes are pairwise disjoint sets is called a *disjoint cover*. If set E_2 is empty, then the cover $COV(E_1|E_2) = COV(E_1|\emptyset)$ is simply denoted as $COV(E_1)$.

Definition 10. The *sparseness* (*the degree of generality*) of a cover is defined as the sum of the sparsenesses (the degrees of generality) of complexes in the cover.

3. SUFFICIENCY OF COMPLEXES AS CLUSTER REPRESENTATIONS

First, we will observe the following property of complexes:

Theorem 4. For any given event space **E** and integer $k \leq d_1 \cdot d_2 \cdots d_n$ (where d_i is the cardinality of the value set of variable x_i), there exist k pairwise disjoint complexes $\alpha_1, \alpha_2, \ldots, \alpha_k$ which completely fill up the space **E**, i.e.,

$$\bigcup_{j=1}^{k} \alpha_j = \mathbf{E} \tag{15}$$

Proof. The theorem is equivalent to saying that any event space can be partitioned into an arbitrary number of complexes (but, of course, not larger than the cardinality of **E**). To see this, take any subset of variables such that the arithmetic product of corresponding d_i's is greater than or equal to k:

$$k \leq k' = \prod_{i \in I} d_i \tag{16}$$

Let R_j, $j = 1, 2, \ldots$ denote all possible sequences of values of variables x_i, $i \in I$. Construct complexes:

$$\alpha_j = \bigwedge_{i \in I} [x_i = r_{ij}] \tag{17}$$

where r_{ij}, $i \in I$, $j = 1, 2, \ldots$, denotes a value of variable x_i in the sequence R_j. Obviously, the complexes α_j are pairwise disjoint and fill up the space **E**. If $k' > k$, then $k' - k$ complexes are joined with the remaining ones into single complexes, according to the following formula:

$$\beta[x_i = a] \vee \beta[x_i = b] \equiv \beta[x_i = a, b] \tag{18}$$

where β denotes a conjunction of selectors involving variables other than x_i. This is always possible, because for any x_i, $i \in I$, there are d_i complexes α_j, which differ only in the value of x_i. ∎

From the viewpoint of clustering, a more interesting question is whether for any given event set E in the space **E**, there always exists an arbitrary number $k \leq c(E)$ of pairwise disjoint complexes, such that they not only fill up the space **E**, but also partition the set E into k nonempty subsets. A positive answer to this question would imply that any given event set can be partitioned into an *a priori* assumed number of subsets, each covered by a simple complex, disjoint from other complexes. The answer is indeed positive. In fact, even a stronger property holds, as stated by the following theorem.

Theorem 5 (The Sufficiency Principle). For an event space **E** and any data event set $E = \{e_1, e_2, \ldots, e_k\}$, $E \subseteq \mathbf{E}$, there exists at least one set of k pairwise disjoint complexes $\alpha_1, \alpha_2, \ldots, \alpha_k$, such that each complex contains one data event:

$$e_j \in \alpha_j, \qquad j = 1, 2, \ldots, k \tag{19}$$

and the union of complexes fills up the space **E**:

$$\bigcup_{j=1}^{k} \alpha_j = \mathbf{E} \tag{20}$$

Proof. The basic idea of the proof is to show that for any $E = \{e_1, e_2, \ldots, e_k\}$, $E \subseteq \mathbf{E}$, it is always possible to construct a tree, in which nodes are assigned the variables x_i, $i \in 1, 2, \ldots, n$, branches of node x_i are assigned elements of a partition of D_i (the value set of x_i), and the leaves represent complexes α_j, such that each complex covers a single event e_j, and the union of complexes fills up the space **E**.

Suppose $e_j = (x_{1j}, x_{2j}, \ldots, x_{nj})$, $j = 1, 2, \ldots, k$, and $x_{ij} \in D_i$.

Take any variable, say x_p, which has different values for events in E. Suppose these values are $a_1, a_2, \ldots, a_z$. Partition the value set, D_p of x_p, into subsets $\{a_1\}, \{a_2\}, \ldots, \{a_{z-1}\}$, A_z, where $a_z \in A_z$ and A_z is a set $D_p \setminus \{a_1, a_2, a_3, \ldots, a_{z-1}\}$. It is obvious that complexes $[x_p = a_1]$, $[x_p = a_2], \ldots, [x_p = A_z]$ partition both the event set E and the event space **E** into z nonempty subsets. Suppose these complexes partition E into E_{a_1}, $E_{a_2}, \ldots, E_{A_z}$ and **E** into $\mathbf{E}_{a_1}, \mathbf{E}_{a_2}, \ldots, \mathbf{E}_{A_z}$, where $E_{a_i} \subseteq \mathbf{E}_{a_i}$.

Variable x_p is assigned to the root of a tree. Branches from the root are assigned values $a_1, a_2, \ldots, A_z$. Leaves of this tree correspond to complexes $[x_p = a_1]$, $[x_p = a_2], \ldots, [x_p = A_z]$, covering event sets $E_{a1}, E_{a2}, \ldots, E_{Az}$, respectively (Fig. 3).

For every one of the above event sets which has more than one element repeat the above process with the following modification. Suppose E_{a1} has more than one element and x_r takes values $b_1, b_2, \ldots, B_y$ for events in E_{a1}. Assign x_r to the root of a new tree, and attach the tree to the leaf

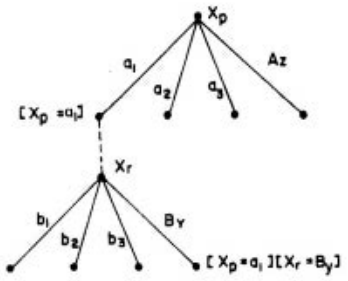

Fig. 3. An illustration for the proof of Theorem 5.

corresponding to E_{a1} (i.e., to the leaf marked by $[x_p = a_1]$ in Fig. 3). Assign the branches emanating from this root values $b_1, b_2, \ldots, B_y$, where $B_y = D_y \setminus \{b_s, b_2, \ldots, b_{y-1}\}$. It is obvious that complexes

$$[x_p = a_1][x_r = b_1], \quad [x_p = a_1][x_r = b_2], \quad \ldots, [x_p = a_1][x_r = B_y]$$

partition both the set E_{a1} and the set Σ_{a1} into y disjoint subsets.

This process is continued until leaves of the obtained tree correspond to complexes, each covering only one event from E. Because every step of this process partitions simultaneously events in E and in $\mathbf{E}$, the union of the obtained complexes covers E and fills up the whole space $\mathbf{E}$. Thus, these complexes constitute the desired set $\{\alpha_1, \alpha_2, \ldots, \alpha_k\}$. ∎

The above theorem asserts that the space of all complexes is sufficient to be a space of cluster representations, as it states that any event set can be clustered into an arbitrary number of complexes. The theorem is used as the theoretical basis for the clustering algorithm described in Sec. 6.

As the above proof indicates, there usually will be many covers which constitute a k-partition of any given set. Therefore, a question arises as to which cover to select as the most desirable. In order to answer this question, a criterion of the quality of a cover is needed.

4. A CRITERION FOR EVALUATING QUALITY OF CLUSTERING

Let E be the set of data points, and $COV(E)$ a disjoint cover of E. Such a cover implies a partition of E into clusters, each cluster being the event set contained in one complex. The sparseness (or the degree of generality) of the cover could be used for defining a criterion of quality of a partition. However, if E is partitioned into individual events, then, obviously, the sparseness (as well as the degree of generality) will be zero. Consequently, this kind of criterion can be used only if the number of clusters is assumed *a priori*, i.e., for a constrained clustering problem. In this case the problem is to find a disjoint cover of E with k complexes, whose sparseness (or the degree of generality) is minimum. In the case of a free clustering problem (i.e., when the number of clusters is not assumed *a priori*), a criterion of quality of partitioning has to involve, in addition to sparseness (or the degree of generality), some "cost" function dependent on the number of clusters, e.g., a measure of complexity of a cover. In this paper we are concerned only with the constraint clustering problem. Although it may seem otherwise, this is not a serious limitation because interesting practical solutions of clustering problems should not produce more than just a few clusters (this is so, because when the number of clusters is large, humans prefer to organize

them into a hierarchy). Consequently, to obtain a general solution, a constraint clustering algorithm should be repeated for several different k, and the best obtained partition selected as the general solution.

The sparseness (or the degree of generality) may not be sufficient as the sole criterion for selecting a cover. One may seek a cover which exhibits other properties than minimum sparseness. In order to use several criteria for selecting a cover simultaneously, we adopt the lexicographic cost functional defined in Ref. 14.

A *lexicographic evaluation functional* (*LEF*) is defined as a pair of two lists:

$$A = \langle a\text{-list}, \tau\text{-list} \rangle$$

where a-list $= (a_1, a_2, \ldots, a_l)$ is a list of attributes to be used to evaluate a cover; and τ-list $= (\tau_1, \tau_2, \ldots, \tau_l)$ is a list of "tolerances" assigned to the attributes a_i, respectively, $0 \leq \tau_i \leq 1$.

Let V_j, $j = 1, 2, \ldots$ denote all possible disjoint covers of the event set E. Let V denote one of the covers, and let $a_i(V_j)$ denote the value of attribute a_i for cover V_j. Cover V is said to be *optimal* (*minimal*) under functional A if for every j

$$A(V) \overset{\tau}{<} A(V_j) \tag{21}$$

where

$$A(V) = (a_1(V), a_2(V), \ldots, a_l(V))$$

$$A(V_j) = (a_1(V_j), a_2(V_j), \ldots, a_l(V_j)), \qquad j = 1, 2, \ldots$$

and $\overset{\tau}{<}$ is a relation, called the *lexicographic order with tolerances*, which holds if

$$\begin{aligned} & a_1(V_j) - a_1(V) > \Upsilon_1 \\ \text{or} \quad & |a_1(V_j) - a_1(V)| \leq \Upsilon_1 \quad \text{and} \quad a_2(V_j) - a_2(V) > \Upsilon_2 \\ & \text{or} \ldots \\ & \vdots \\ \text{or} \ldots \quad & \qquad\qquad \text{and} \quad a_l(V_j) - a_l(V) \geq 0 \end{aligned} \tag{22}$$

where

$$\Upsilon_i = \tau_i \cdot (a_{i\max} - a_{i\min}), \qquad i = 1, 2, \ldots, l-1$$

$$a_{i\max} = \max_j \{a_i(V_j)\}$$

$$a_{i\min} = \min_j \{a_i(V_j)\}$$

Note that if $\tau = (0, 0, \ldots, 0)$ then $\overset{\tau}{<}$ denotes the lexicographic order in the usual sense. In this case, A can be specified just as $A = \langle a\text{-list} \rangle$.

To specify a functional A one selects a set of attributes, puts them in the desirable order in the a-list, and sets the values for tolerances in the τ-list.

Relation $\overset{\tau}{<}$ partitions all covers into equivalence classes and orders the classes linearly, with the first class containing one or more optimal covers, and the next classes containing consecutively less optimal covers.

Below are a few criteria which may be used to assemble an a-list:

- *Sparseness* (or generality g) of a cover. Minimizing sparseness will produce complexes which "fit" as closely as possible to clusters of data events. This criterion is an analogue to the criterion of minimizing intradistances in the conventional distance-based clustering.

- *Intersection*, defined as the average degree of intersection (DI) between any two complexes in the cover. The DI between two complexes is the total number of selectors which remain in both complexes after removal of every pair of disjoint selectors (selectors whose reference sets do not intersect). For example, the degree of intersection between complexes

$$[x_2 = 2, 3][x_4 = 3, 5, 7][x_5 = 2 \ldots 5]$$

and

$$[x_1 = 3][x_2 = 1][x_4 = 5 \ldots 12][x_5 = 1]$$

is 3. ($\rightleftarrows$ links disjoint selectors.)

The introduction of DI as a criterion for clustering comes from the observation that people tend to prefer partitions of objects in which clusters differ not in just one, but in many characteristics. This criterion is an analogue to the criterion of maximizing cluster interdistances in distance-based clustering.

- *Imbalance*, defined as

$$1/k \sum_{i=1}^{k} |1/k \cdot c(E) - c(E \cap \alpha_i)| \tag{23}$$

where $c(E)$ is the size of the event set, and $c(E \cap \alpha_i)$ is the number of data events covered by complex α_i (the cardinality of the core of α_i). The imbalance measures the variability of cluster sizes.

- *Dimensionality*, defined as the total number of different variables involved in the complexes of the cover. The dimensionality tells us how many variables are used to describe clusters, and, thus, how many variables have to be measured to classify objects into these clusters.

5. PROCEDURES STAR AND NID

Before describing an algorithm for conceptual clustering (Sec. 6) we shall describe two important procedures used in this algorithm: STAR and NID. Procedure STAR generates the star (Definition 8) of a data event against a set of other data events, and procedure NID transforms a nondisjoint cover, whenever possible, into a disjoint cover with the same number of complexes.

5.1. Procedure STAR

This procedure is based on the algorithm described in Ref. 14.

Let e_0 be an event and α a complex. The operation $e_0 \vdash \alpha$ (read: e_0 *extended in* α) is defined as follows:

$$e_0 \vdash \alpha = \begin{cases} \alpha, & \text{if } e_0 \in \alpha \\ \varnothing, & \text{otherwise} \end{cases} \tag{24}$$

Let event $e_1 = (r_1, r_2, \ldots, r_n)$ and $e_1 \neq e_0$. The operation $e_0 \dashv e_1$ (read: e_0 *extended against* e_1) is defined as follows:

$$e_0 \dashv e_1 = \bigcap_{i \in I} (e_0 \vdash [x_i \neq r_i]) \tag{25}$$

Let $G^u(e \mid E)$ denote the union of complexes from the star $G(e \mid E)$. It can be shown that

$$G^u(e \mid E) = \bigcap_{e_j \in E} (e \dashv e_j) \tag{26}$$

To obtain the star $G(e \mid E)$ from $G^u(e \mid E)$, the right-hand side of (26) must be converted to the union of maximal (under inclusion) complexes. Such a union is obtained when the set-theoretical multiplication is done with the application of absorption laws.

5.2. Procedure NID

This is transformation of a *n*on-disjoint cover *i*nto a *d*isjoint cover.)

Let $\{\alpha_1, \alpha_2, \ldots, \alpha_l\}$ be a set of not necessarily disjoint complexes, which is a cover of a data event set F.

1. Let $c(\alpha_i)$, $i = 1, 2, \ldots, l$, denote the cardinality of α_i (the total number of events covered). Determine the (arithmetic) *s*um of *c*ardinalities:

$$sc = \sum_{i=1}^{l} c(\alpha_i) \tag{27}$$

and the *c*ardinality of the (set-theoretic) *s*um of complexes:

$$cs = c\left(\bigcup_{i=1}^{l} \alpha_i\right) \tag{28}$$

2. If $sc = cs$ then STOP: L is already a disjoint cover.
3. For $i = 1, 2, \ldots, l$, determine the relative core, CORE_i, of complex α_i, i.e., the set containing data events covered by complex α_i and only by this complex. Let RESIDUE denote the set of remaining events, i.e., $\text{RESIDUE} = F \setminus \bigcup_{i=1}^{l} \text{CORE}_i$.
4. For each CORE_i determine its mc-complex (Definition 6):

$$\alpha_i^0 = RU(\text{CORE}_i), \qquad i = 1, 2, \ldots, l$$

5. If any two complexes α_i^0 intersect, then STOP. The disjoint cover cannot be obtained. (This is a direct consequence of Theorem 1.)
6. Select an event from RESIDUE and call it e. Delete e from RESIDUE.
7. For each pair (e, α_i^0), $i = 1, 2, \ldots, l$, determine the covering complex:

$$\alpha_i^1 = RU(\{e\} \cup \alpha_i^0)$$

8. Delete every α_i^1 which intersects with any α_j^0, $j \neq i$. If all α_i^1 are deleted then STOP: a disjoint cover cannot be obtained.
9. Select the best complex, Best-α, among remaining α_i^1, according to the LEF:

$$\langle(\Delta\text{spars}, -\text{res}, -\Delta\text{sel})(\tau_1, \tau_2, \tau_3)\rangle$$

where

Δspars = the difference between the sparseness of α_i^1 and α_i^0,

res = the number of events in RESIDUE covered by α_i^1,

Δsel = the difference between the number of selectors in α_i^0 and α_i^1,

τ_1, τ_2, τ_3 = tolerances set to 0 by default.

The sign "$-$" in front of res and Δsel indicates that the algorithm will maximize these criteria (by minimizing the negative value).

10. Suppose Best-α was created by joining e with α_b^0. Assume that α_b^0 denotes now Best-α.
11. If $\text{RESIDUE} = \varnothing$, then END, otherwise go to 6.

The output from this procedure is either a disjoint cover $\{\alpha_1^0, \alpha_2^0, \ldots, \alpha_l^0\}$ of set F, or an indication that such cover cannot be obtained from the initial cover $\{\alpha_1, \alpha_2, \ldots, \alpha_l\}$.

6. AN ALGORITHM FOR CONJUNCTIVE CONCEPTUAL CLUSTERING

6.1. An Overview

Based on the ideas described in previous sections, we have developed an algorithm for conjunctive conceptual clustering, called PAF.[5] Given a set, E, of events from an arbitrary event space, and an integer k, PAF partitions E into k clusters, each of which has a conjunctive description in the form of a VL_1 complex. The obtained partition is optimal or suboptimal with regard to a lexicographic evaluation function, assembled by a user from the criteria listed in Sec. 5.

The general structure of the algorithm is based on the multicriteria dynamic clustering method developed by Diday and his collaborators (see the work of Diday and Simon[1] and Hanani[15]). Underlying notions of the dynamic clustering method are two functions:

- g—the *representation function*, which, given k clusters of a partition of E (a k-partition) produces a set of k cluster representations, called *kernels*. There may be different kinds of kernels, e.g., the center of gravity of a cluster, a few selected points from a cluster, a probability distribution best fitting the cluster, a linear manifold of minimal inertia, or a VL_1 complex.
- f—the *allocation function*, which, given a set of kernels, partitions E into k clusters, "best fitting" these kernels.

The method works iteratively, starting with a set of k initial, randomly chosen kernels (of a given kind). A single iteration consists of an application of function f to given kernels, and then of function g to the obtained partition. An iteration ends with a new set of kernels. The process continues until the chosen criterion of quality of a partition, W, ceases to improve. (Criterion W measures the "fit" between a partition and kernels.) It has been proven[1] that this method always converges to a local optimum.

The measure W can be a single criterion or a sequence of criteria. In the multicriteria case, for each criterion an appropriate type of kernels is used (see Hanani[15]).

The algorithm PAF applies a multicriteria dynamic clustering method, in which the basic and final cluster representation is a VL_1 complex.

[5] Polish–American–French.

Intermediate representation include the geometrical center of a cluster (using the syntactic distance; Definition 1) and the "most outstanding" event (most distant from the center) in a cluster.

The use of the latter representation is an application of an "adversity principle." This principle states that if the most outstanding event truly belongs to the given cluster, then if it serves as the cluster representation, then the "fit" between it and other events in the same cluster should still be better than the "fit" between it and events of any other cluster.

The algorithm PAF does not use any conventional distance function (or other measure of "fit") between a data event and a kernel (in this case a VL_1 ccomplex). An event either belongs to a complex or does not belong. A complex is a form, which can describe a very large number of configurations of events. For n variables, each taking d distinct values, there are $N = (2^d - 1)^n$ different complexes. For example, if $n = 10$ and $d = 7$, then $N \simeq 10^{20}$. Such a large size of the "concept space" makes conjunctive clustering computationally an extremely complex problem. To obtain a feasible practical solution, it is necessary to apply a combination of carefully designed heuristic search methods. In PAF, one of the methods used is a well-known "best first" search technique developed in artificial intelligence.[16]

6.2. Description of PAF

A flow diagram of algorithm PAF is shown in Figure 4.

1. In the first step (block 1), a set of k data events $E_0 = \{e_1, e_2, \ldots, e_k\}$, called *seeds*, is selected from the event set E. Seeds can be selected arbitrarily, or they can be chosen as events which are most distant syntactically (Definition 1) from each other. In the latter case the algorithm will generally converge faster. For selecting such events program ESEL[17] can be used.

2. For each seed e_i, $i = 1, 2, \ldots, k$, a star is generated against the remaining seeds (using procedure STAR described in Sec. 5):

$$G_i = G(e_i | (E_0 \backslash \{e_i\})), \qquad i = 1, 2, \ldots, k$$

3. From each star a complex is selected, such that the resulting set of k complexes:
   a. Is a disjoint cover of E.
   b. Is an optimal or suboptimal cover among all possible such covers, according to an assumed criterion LEF (constructed by a user from criteria listed in Sec. 4: sparseness or generality, intersection, imbalance, and dimensionality). This is the most difficult and computationally costly step of the algorithm. It can be

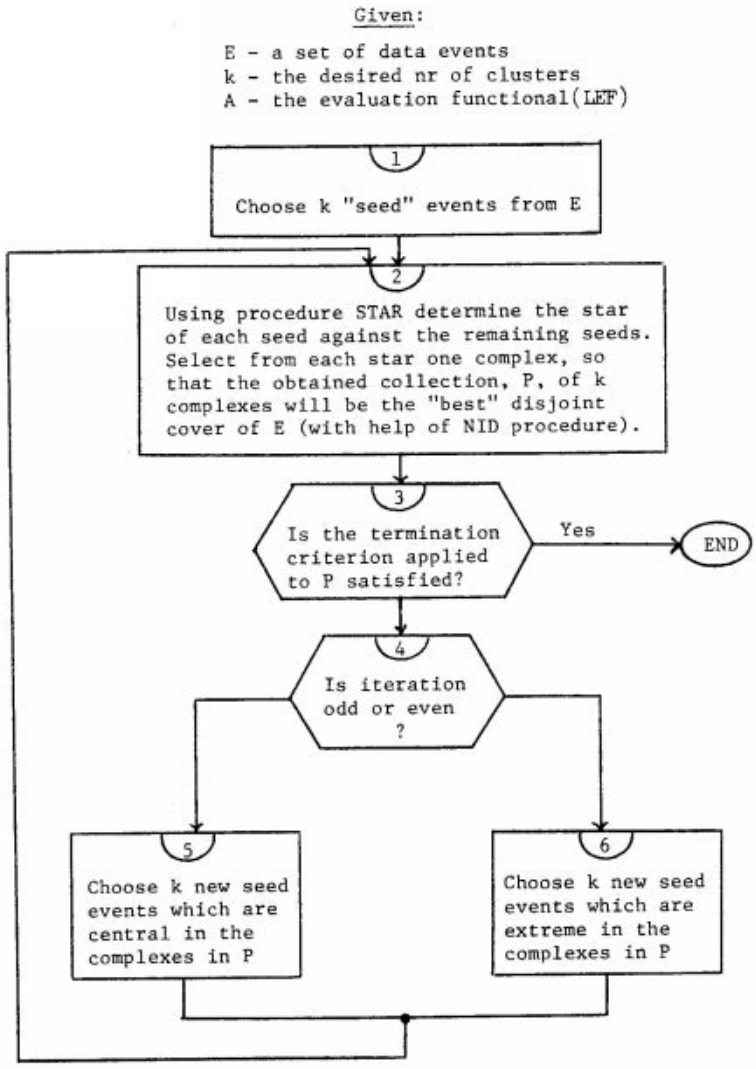

Fig. 4. A flow diagram of algorithm PAF.

performed in a number of different ways. We will distinguish between three different procedures: P (parallel), PS (parallel-sequential), and S (sequential). These procedures are described in Sec. 7.

4. A termination criterion of the algorithm is applied to the obtained cover. The termination criterion is a pair of parameters (b, p), where b (the *b*ase) is a standard number of iterations the algorithm always performs, and p (the *p*robe) is the number of iterations beyond b

which the algorithm performs after each iteration which produces an improved cover.

5. A new set of seeds is determined. If the iteration is odd, then the new seeds are data events in the centers of complexes in the cover (according to the syntactic distance). If the iteration is even, then the new seeds are data events maximally distant from the centers (according to the "adversity principle").

7. PROCEDURES P, SP, AND S

All three procedures use *bounded stars*, that is, stars whose size is limited by special parameter MAXSTAR. The reason is that the size of stars may be very large when the number of variables n is high. As can be seen from procedure STAR, the upper bound on the number of complexes in a star grows exponentially with k (the number of clusters); namely n^k. The size of any star is controlled by not allowing it to have more than MAXSTAR complexes. Whenever a star exceeds this number, complexes are ordered in the order of ascending sparseness, and only first MAXSTAR complexes are retained. It is also assumed that all complexes in stars are trimmed (i.e., the refunion operation is applied to the core of each complex, and then the resulting mc-complex is used to replace the original complex in the star; see Definition 7).

To simplify the description of procedures we will assume that the criterion of clustering optimality is minimizing the sparseness of the disjoint cover (representing a partition). The procedures can be extended for a multicriteria case by using a criterion LEF (which imposes a linear order between equivalence classes of sets of complexes). In such a multicriteria case, however, sparseness should be used as the primary criterion in order to retain the properties of the described procedures.

7.1. Procedure P

The procedure is applicable for relatively small MAXSTAR and k. It is particularly useful for executuion on a parallel processor. Let star $G_i = G(e_i|(E_0\backslash\{e_i\}))$ be a set $\{\alpha_0^i, \alpha_1^i, \ldots, \alpha_{g_i}^i\}$, $i = 1, 2, \ldots, k$. Assume that complexes α_j^i, $j = 0, 1, \ldots, g_i$, are ordered in ascending order on sparseness. The position of a complex in the star so ordered (indicated by a subscript, which counts from 0) is called the *rank* of the complex (thus, e.g., complex α_2^i has rank 2).

Taking one symbol α_j^i from each star G_i, $i = 1, 2, \ldots, k$, at a time, generate all possible sequences:

$$\begin{aligned} P_0 &= (\alpha_0^1, \alpha_0^2, \ldots, \alpha_0^k) \\ P_1 &= (\alpha_0^1, \alpha_0^2, \ldots, \alpha_0^{k-1}, \alpha_1^k) \\ &\vdots \\ P_{g_k} &= (\alpha_0^1, \alpha_0^2, \ldots, \alpha_{g_k}^k) \\ &\vdots \\ P_\Gamma &= \{\alpha_{g_1}^1, \alpha_{g_2}^2, \ldots, \alpha_{g_k}^k\} \end{aligned} \tag{29}$$

where $\Gamma = (g_1 + 1)(g_2 + 1) \cdots (g_k + 1)$.

The sum of the ranks of complexes in any such sequence is called the *pathrank*. It is assumed that sequences P_j, $j = 1, 2, \ldots, \Gamma$, are generated in the ascending order on their pathrank. Thus, P_0 has the pathrank 0 (because all complexes in P_0 have rank 0); $P_2, P_3, \ldots, P_{k+1}$ have pathrank 1; and P_Γ has pathrank $g_1 + g_2 + \cdots + g_k$. The order of sequences with the same pathrank is irrelevant.

Considering sequences P_j in the ascending order on their pathrank, the following operations are performed on each sequence:

1. A P_j is tested whether it is a cover of E. This can be done by consecutively removing from E data events covered by each complex in P_j. If at the end E becomes the empty set, P_j is a cover. If a P_j is not a cover, it is removed from further consideration.
2. A P_j is tested whether it is a disjoint cover. If it is, its sparseness is calculated. If it is not, a lower bound (l.b.) on the sparseness of a possible disjoint cover is calculated (without actually determining the disjoint cover). The l.b. is computed by determining the relative core of each complex (i.e., data events covered only by the given complex and not by any other complexes), and then computing the sparseness of the mc-complex of the core. The l.b. is the sum of so obtained sparsenesses (this computation is based on Theorem 3). {The purpose of using the l.b. is to avoid, whenever possible, the computationally costly procedure NID.}
3. If the computed sparseness (or l.b.) is not a new minimum (i.e., is not smaller than the sparseness of the best cover obtained so far), then the cover is removed from further consideration. Otherwise, if it is a disjoint cover, it is retained as the best cover; and if it is a nondisjoint cover, it is transformed by NID, if possible, into a disjoint cover (note that some operations of the NID procedure were already done

in step 2). If the sparseness of the obtained disjoint cover still represents a new minimum, the cover is retained as the best so far. If the sparseness is not a new minimum, or NID fails to produce a disjoint cover, the cover is removed from further consideration.

The disjoint cover retained at the end of the above search process through sequences P_j is the output of the procedure. It is a minimum sparseness cover which can be assembled from complexes in the given stars. The existence of at least one disjoint cover is assured by the sufficiency principle.[6] An advantage of the above-described ordering of sequences P_j is that the best cover will most likely be close to the beginning of the list. Therefore, if the number of sequences is very large, the search can stop before reaching the end, with a low risk of losing the optimal solution.

7.2. Procedure PS

In procedure P, all sequences P_j were generated first, and then linearly searched in order to determine the best cover. In this procedure, the search for the best cover is done during the process of generating the sequences, using the "best first" search strategy (see Winston[16]). Specifically, the search is based on the algorithm A^* (see Nilsson[18]). At each step a complex is added to the partial cover (a partial sequence after application of NID) which most likely leads the optimal cover (according to an evaluation function). This process avoids testing (usually many) sequences P_j, for which it is possible to predict that they will not produce an optimal cover. The procedure PS is especially applicable when stars G_i are large.

Figure 5 illustrates the search process. Branches of the tree at level i represent complexes in star G_i. A path from the root to a node at level i represents a partial disjoint cover with i complexes. When $i = k$, the path represents a complete disjoint cover (corresponding to some sequence P_j to which NID was applied).

In the first step, sequence $P_0 = (\alpha_1^2, \alpha_2^0, \ldots, \alpha_k^0)$ is generated. (It is the sequence of complexes of the smallest sparseness.) The relative core of each complex is determined and then the mc-complex is constructed for each core. Let $s_1, s_2, \ldots, s_k$ denote the sparsenesses of the obtained mc-complexes. On the basis of Theorem 3, the sum $s_1 + s_2 + \cdots + s_k$ specifies a lower bound on the sparseness of the best disjoint cover which can be built from complexes of given stars.

In the next step, node ① (Fig. 5) is expanded, i.e., α_1^0 is paired with every complex in G_2, procedure NID is applied to each pair, and then the sparseness is calculated for the obtained disjoint pair. If NID fails, the path is abandoned. The obtained pair is a partial cover with $i = 2$ complexes. Nodes

[6] Assuming sufficiently large MAXSTAR.

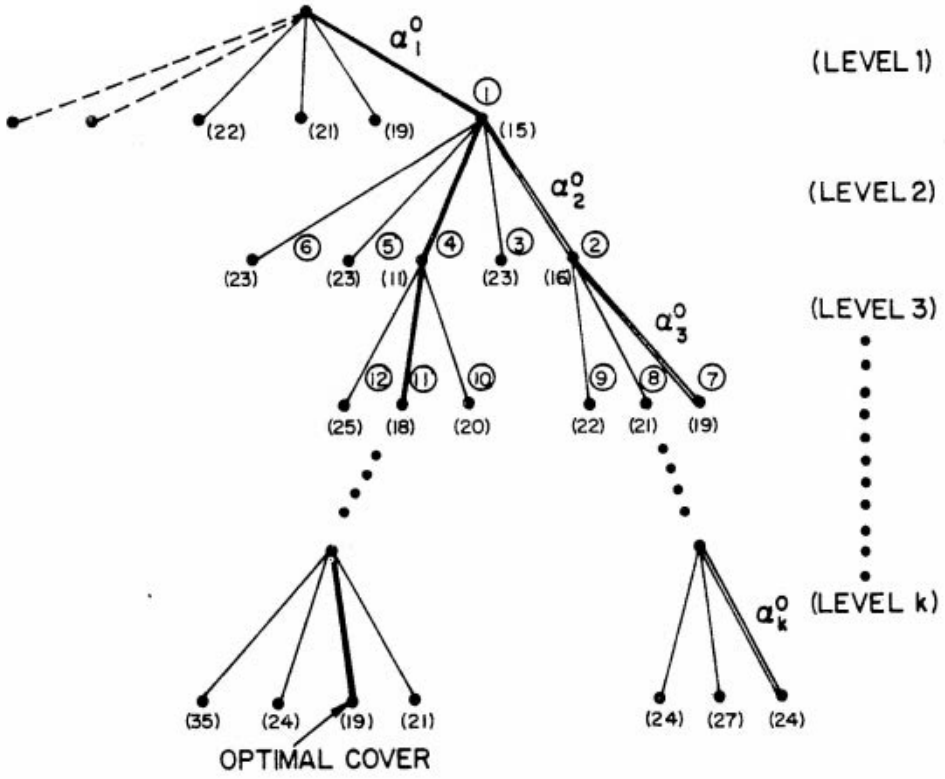

Fig. 5. A search tree illustrating procedure PS.

corresponding to generated partial covers (including the remaining complexes in G_1) are assigned a value of the evaluation function:

$$f = h + g \tag{30}$$

where h is the sparseness of the obtained partial disjoint cover, and g is the sum $s_{i+1} + s_{i+2} + \cdots + s_k$, where i is the number of complexes in the partial cover. {g represents a l.b. on the sparseness of the remaining complexes to be determined, i.e., complexes that are needed to complete the cover under construction.}

According to the best first strategy, the node to be expanded at each step is the one which is associated with the lowest value of the evaluation function. It has been proved that such strategy will produce the optimal cover.[18] The order of expanding nodes in the tree in Fig. 5 is shown by numbers in circles. The value of the evaluation function associated with each node is given in parentheses.

7.3. Procedure S

This procedure is like procedure PS, with the exception that stars are not generated beforehand. When expanding a node in the search tree, rather than taking complexes from already determined stars, an appropriate star is generated each time. This requires a multiple repetition of the star generation process, but saves on the memory for storing all stars (which may be large sets).

8. A NOTE ON IMPLEMENTATION AND AN EXAMPLE

The algorithm has been implemented by R. Stepp in PASCAL for Cyber 1975. The details on the implementation are in Ref. 19. For illustration, we will present here a simple example, representing one of the testing experiments with the program.

Figure 6(a) represents a diagrammatic representation[20] of an event space, spanned over variables x_1, x_2, x_3, x_4, with domain sizes 2, 5, 4, 2 respectively. Each cell represents one event. Cells marked by 1 represent data events; remaining cells represent empty events. Figure 6(a) also shows a cover obtained from the first iteration of the algorithm. The remaining figures show results from the consecutive iterations. Cells representing seed events in each iteration are marked by +. The partition evaluation criterion was a LEF:

$$\langle(\text{sparseness, imbalance, dimensionality})\ (0, 0, 0)\rangle$$

According to this criterion, the best partition is the one shown in Fig. 6(c). The partition is specified by complexes:

$$\alpha_1^0 = [x_1 = 0][x_2 = 1][x_4 = 0]$$

$$\alpha_2^0 = [x_1 = 0][x_2 = 2][x_3 = 1\,..\,3]$$

$$\alpha_3^0 = [x_1 = 1][x_2 = 1\,..\,3]$$

Another experiment with the program involved clustering 47 cases of soybean diseases. These cases represented four different diseases, as determined by plant pathologists (the program was not, of course, given this information). A single case was specified by an event of 35 many valued variables. With $k = 4$, the program partitioned all cases into four categories. These four categories turned out to be precisely the categories corresponding to individual diseases. The complexes defining the categories involved known characteristic symptoms of the corresponding diseases.

Thus, the program not only correctly partitioned the events, but also found a description of each cluster, compatible with the corresponding human description.

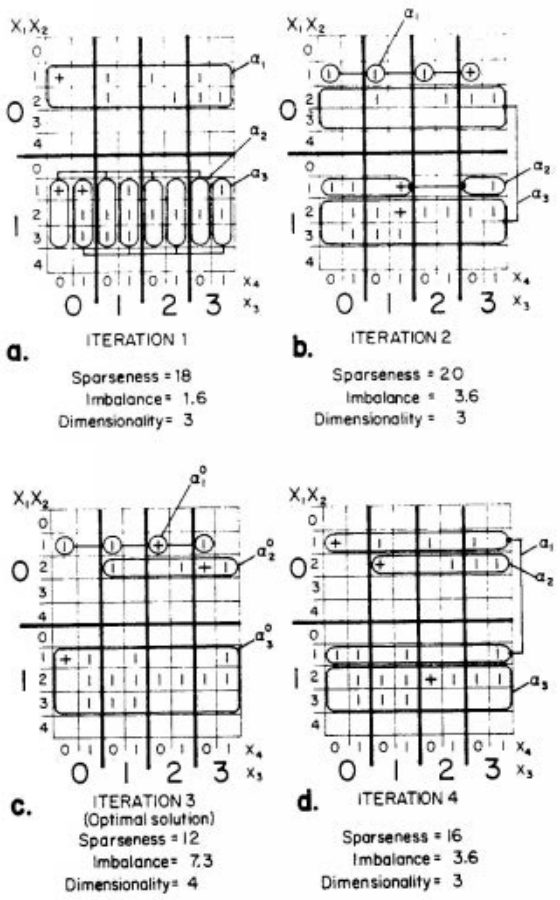

Fig. 6. An example of consecutive iterations.

9. CONCLUSION

This paper described a theoretical foundation and an algorithm for conceptual clustering, in which entities are assembled into classes described by conjunctive concepts (VL_1 complexes). Thus, the presented approach produces not only clusters but also their descriptions (unlike any conventional clustering method). The descriptions are conjunctions of relations on variables characterizing the entities, and thus have a simple conceptual interpretation.

The presented algorithm has been implemented and tested on various examples. The results indicate that the method provides a radically new alternative to the conventional clustering methods, and has a potential for novel solutions of a variety of clustering problems.

ACKNOWLEDGMENTS

A part of the research reported in this paper was done when the author worked as a Visiting Professor at the University of Paris—IX (Dauphine) and at the research institute IRIA in France. A partial support provided by these institutions, and by the National Science Foundation under Grant MCS-79-06614, is gratefully acknowledged. Numerous conversations with Professor E. Diday and his collaborators at IRIA have been very useful in shaping up the ideas and the method described here. Some of the results related to the algorithm PAF have been developed in collaboration with Bob Stepp. His suggestions and insightful criticisms contributed significantly to the final version of the paper.

REFERENCES

1. E. Diday and J. C. Simon, "Clustering analysis," in *Communication and Cybernetics*, Vol. 10, K. S. Fu, Ed. (Springer-Verlag, Berlin, Heidelberg, New York, 1976).
2. R. Anderberg, *Cluster Analysis for Applications* (Academic Press, New York, 1973).
3. E. Becker, "Cluster Analysis Formalized as a Process of Fuzzy Identification Based on Fuzzy Relations," Report IT-78-15, Department of Electrical Engineering, Delft University of Technology, The Netherlands (October 1978).
4. K. Gowda and G. Krishna, "Disaggregative clustering using the concept of mutual nearest neighborhood," *IEEE Trans. Systems, Man and Cybernetics*, **SMC-8**(12): 888–894 (December 1978).
5. R. S. Michalski, "Studies in Inductive Inference and Plausible Reasoning" a research proposal to NSF, November 1978), funded under Grant MCS-79-06614.
6. R. Stepp, "Learning Without Negative Examples via Variable-Valued Logic Characterizations: The Uniclass Inductive Program AQ7UN1," Report 982, Department of Computer Science, University of Illinois, Urbana, Illinois (July 1979).
7. S. Watanabe, "Pattern recognition as an inductive process," in *Methodologies of Pattern Recognition* Watanabe, Ed. (Academic Press, New York, 1968).
8. S. Watanabe, *Knowing and Guessing: A Quantitative Study of Inference and Information* (Wiley, New York, 1969).
9. D. W. Matula, "Cluster analysis via graph-theoretic techniques," *Proceedings of the Louisiana Conference on Combinatorics, Graph Theory and Computing*, R. C. Mullin, K. B. Reid, and D. P. Roselle, Eds. Louisiana State University, Baton Rouge, Louisiana (March 1–5, 1970).
10. I. G. Auguston and J. Minker, "An analysis of some graph theoretical cluster techniques," *J. ACM* **17**(4):571–588 (October 1970).
11. C. T. Zahn, "Graph-theoretic methods for detecting and describing Gestalt clusters," *IEEE Trans. Comps.* **C-20**(1):68–86 (January 1971).
12. C.-M. Cheng, "Clustering by Clique Generation," Report 655, Department of Computer Science, University of Illinois, Urbana, Illinois (June 1974).
13. R. S. Michalski, "Variable-valued logic: System VL_1," *Proceedings of the 1974 International Symposium on Multiple-Valued Logic*, West Virginia University, Morgantown, West Virginia (May 29–31, 1974).
14. R. S. Michalski, "Synthesis of optimal and quasi-optimal variable-valued logic formulas," *Proceedings of the 1975 International Symposium on Multiple-Valued Logic*, Bloomington, Indiana (May 13–16, 1975).
15. U. Hanani, "Multicritera Dynamic Clustering," Reports of IRIA, France (1979).

16. P. H. Winston, *Artificial Intelligence* (Addison-Wesley, Reading, Mass., 1977).
17. R. S. Michalski and J. B. Larson, "Selection of Most Representative Training Examples and Incremental Generation of VL_1 Hypotheses: The Underlying Methodology and the Description of Programs ESEL and AQ11, Report No. 867, Department of Computer Science, University of Illinois, Urbana, Illinois (1978).
18. T. Nilsson, *Principles of Artificial Intelligence* (Tioga Publishing Company, 1980).
19. R. Stepp, "A Description and User's Guide for CLUSTER/PAF—A Program for Conjunctive Conceptual Clustering," to appear as a report of the Department of Computer Science, University of Illinois, Urbana, Illinois (1980).
20. R. S. Michalski, "A Planar Geometrical Model for Representing Multidimensional Discrete Spaces and Multiple-Valued Logic Functions," Report No. 897, Department of Computer Science, University of Illinois, Urbana, Illinois (1978).