A Hierarchical Conceptual Clustering Algorithm

Douglas Fisher

Department of Information and Computer Science

University of California at Irvine

Irvine, California 92717

March 1984 CS-21

Technical Report 85-21

(submitted July 1985)

A Hierarchical Conceptual Clustering Algorithm

Abstract

An algorithm is described which constructs a hierarchical taxonomy over object sets. The algorithm falls within the purview of conceptual clustering, and is computationally more efficient than conceptual clustering algorithms currently reported in the literature.

1. Introduction

Given a set of objects (events, observations), clustering is the process of constructing a classification scheme over the object set. The classification scheme serves to divide the object set into distinct classes (subsets), so as to optimize the classification with respect to some clustering criteria. In methods of numerical taxonomy [1], object grouping is based on a numeric measure of object distance or similarity (roughly the inverse of distance). A distance or similarity function is applied to symbolic object descriptions, and object grouping strives to maximize inter-cluster distance, and minimize intra-cluster object distance. Recently, a number of researchers in machine learning have developed what shall be termed conceptual clustering algorithms[1] (eg. [7, 2]). Conceptual clustering algorithms differ from methods of numerical taxonomy in two important respects.

1. Conceptual clustering algorithms return characteristic.

---

[1] A term apparently due to Michalski [5].

(or summary) descriptions of object groupings, whereas methods of numerical taxonomy do not.

2. Conceptual clustering algorithms strive to optimize object clusters according to criteria imposed at the characteristic description level (eg. the 'simplicity' of characteristic descriptions of object groups), and/or the map between characteristic descriptions and the objects they describe (eg. the degree of generality).

This paper presents a hierarchical conceptual clustering program, RUMMAGE (implemented in SIMULA), which forms a decision tree over an object set.

In the following section the language of object and characteristic descriptions is introduced. Section 3 gives a high-level description of the RUMMAGE clustering algorithm. Section 4 details the clustering criteria used by RUMMAGE and how they guide decision tree construction. Section 5 introduces a taxonomy of conceptual clustering algorithms, which is used to facilitate comparisons between RUMMAGE and previous work. The paper concludes with possible extensions to the algorithm.

2. Definition of Terms

The language of object description and object group characteristic descriptions is adopted from Michalski and Stepp [7]

A **variable** is a dimension along which an object may be described (eg. color, size, or shape). The domain of a variable, V, denoted DOMAIN[V], is the set of all values which may instantiate V. The current implementation of RUMMAGE allows only

nominal variables (ie. the domain of which is a finite set of unordered values).

Given a non-empty set of objects, O, the set of values of a variable, V, over O is termed the **reference** to V with respect to O. We indicate a value set is a **reference** to a variable V, by writing a **selector** of the form [V,<ref>] where <ref> is a reference to V. The selector, [V,{f1, f2, ..., fn}] may be read as the variable V may equal f1 or f2 or ... or fn. A selector, S, then has two fields which may be accessed by writing VAR[S] and REF[S], which indicate the variable and reference parts of S, respectively.

A characteristic description of a set of objects is a conjunction of selectors with distinct variable parts, and is termed a **complex**. An **object** is a complex in which the reference part of each selector is a singleton set.

3. Overview of the Algorithm

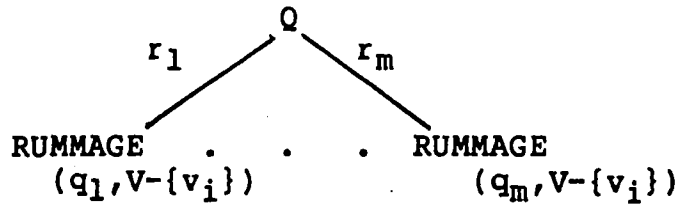
RUMMAGE constructs classifications in the form of decision trees over an object set. Decision tree construction proceeds top-down in a depth-first manner. At any node, N, in the expanding tree RUMMAGE divides the objects represented by node N into mutually-exclusive subsets which will be represented by children of N. Arcs from a node to its children are labeled by rules which distinguish objects of a given child from objects of all other children. Given object symbolic descriptions are defined over a variety of variables (eg. color, size), an

arc-labeling rule formed by RUMMAGE is a single selector. For any node, N, in the decision tree, all arcs to the children of N are labeled by single selectors with the same variable parts and non-intersecting reference parts.

Input to RUMMAGE includes a set of objects, Q, and a set of variables, V, over which objects are defined. RUMMAGE selects that variable, v_i, whose values 'best' partition Q into mutually-exclusive subsets, q_1 through q_m, with respect to clustering criteria to be discussed in the following section. RUMMAGE is then recursively called for each subset, q_j. Recursion bottoms-out when RUMMAGE decides there is no variable which produces a partition of some required minimal quality. Figure 3-1 gives a high-level description of the algorithm.

```
FUNCTION RUMMAGE(Q, V)
BEGIN
  select a variable, v_i, for which there exists
    a partitioning of DOMAIN[v_i], r_1 through r_m,
    which implies a 'best' partitioning of Q, q_1
    through q_m.

  IF no v_i is selected THEN RETURN Q
    ELSE RETURN
```




```
END
```

Figure 3-1: A High-level Description of RUMMAGE

In the following section the process of variable selection, and the clustering criteria used to guide this process are further explicated.

4. Selecting a Variable for Node Division

At any node in the expanding tree, RUMMAGE needs to select a variable whose values best partition an object set with respect to some clustering criteria. The gist of variable selection is to choose that variable, whose values imply the object groups with the best characteristic descriptions over the remaining variables. For example, consider the following objects, defined over variables V_1, V_2, and V_3.

$O_1 = \{ [V_1,\{f_{11}\}], [V_2,\{f_{21}\}], [V_3,\{f_{31}\}] \}$
$O_2 = \{ [V_1,\{f_{11}\}], [V_2,\{f_{21}\}], [V_3,\{f_{32}\}] \}$
$O_3 = \{ [V_1,\{f_{12}\}], [V_2,\{f_{22}\}], [V_3,\{f_{31}\}] \}$
$O_4 = \{ [V_1,\{f_{12}\}], [V_2,\{f_{22}\}], [V_3,\{f_{32}\}] \}$

The values of V_1 and the complexes over V_2 and V_3 they logically imply can be written:

f_{11} ---> $\{f_{21}\}$ and $\{f_{31}$ or $f_{32}\}$
f_{12} ---> $\{f_{22}\}$ and $\{f_{31}$ or $f_{32}\}$

Suppose we wish to select that variable whose values imply complexes with the fewest number of terms (corresponding perhaps to the simplicity of the implied complex), then V_1 or V_2 would be selected to divide O_1 through O_4, rather than V_3 whose values imply more 'complicated' complexes.

For each variable, an implication is constructed for each value of the variable. The implications for each variable are placed as an entry in an implication table. RUMMAGE then

combines implications of the same table entry whose right-hand sides possess a sufficiently large proportion of values in common. The proportion of shared values necessary for combination of implications is supplied by the user. The proportion of values common to two selectors, S_i and S_j with $VAR[S_i] = VAR[S_j]$ is given by[2]

$$\frac{| REF[S_i] \cap REF[S_j] |}{| REF[S_i] \cup REF[S_j] |} \triangleq P_{ij}$$

The proportion of values shared by two complexes (ie. the right-hand sides of two implications) is given by the average over all P_{ij}, where i and j indicate selectors S_i and S_j with identical variable parts. The average over all P_{ij} represents the degree of similarity between two complexes and is denoted Sim_{kl}, for two complexes, C_k and C_l. If all table entries contain a single implication, following implication combination, then RUMMAGE concludes that there is no set of rules which would divide the object set in a non-arbitrary way and no variable is selected for dividing the object set.

After the implication combination process has occurred for all implication table entries, each entry is evaluated along two criteria. The left-hand sides of implications of that table entry which is evaluated as 'best' are used as divisive rules for node

---

[2]If S is a set then |S| denotes the size of S. The symbol, $\triangleq$, may be read "is denoted by".

division. The clustering criteria used by RUMMAGE are given below.[3]

1. **Simplicity** - The simplicity of a complex (ie. the right-hand side of an implication) is a function of the size of the reference part of each selector and is given as

   $$1/(\text{average size of the reference parts of all selectors in C})$$

   The average simplicity over all implications of a table entry, E, is denoted as Ave_simp_E.

2. **Inter-cluster difference** - The difference between two complexes, C_i and C_j, is given as $1 - Sim_{ij} = Diff_{ij}$. The degree of inter-cluster difference exhibited over a set of complexes is the average over $Diff_{ij}$ for all distinct complexes, C_i and C_j, and is denoted $Inter\_diff_E$, for an entry, E. The implication combination process discussed above, insures that any set of rules selected imply complexes possessing a user controlled minimum degree of inter-cluster difference.

The importance of each of these criterion in selecting that variable whose values best partition the object set, may be weighted via user inputed parameters, u and v. The table entry, E, for which the following value is maximized supplies the divisive rules for node expansion.

$$(u * Inter\_diff_E) + (v * Ave\_simp_E)$$

$$\text{where } u + v = 1$$

We now demonstrate the method with an example.

---

[3]The criteria used are a subset of those general criteria suggested by Michalski and Stepp [7] but differ in specifics.

4.1. An Example

Suppose the following variables and their respective domains are used to describe animals.

Variable, V	DOMAIN[V]
Body covering	hair,feathers,cornified skin, moist skin
# of heart chambers	4, imperfect 4, 3
Body temperature	regulated, unregulated
Fertilization	internal,external

Animals (objects) are as follows

Body Covering	Heart Chambers	Body Temperature	Fertilization
hair	4	regulated	internal
feathers	4	regulated	internal
corn. skin	imp. 4	variable	internal
moist skin	3	variable	internal
moist skin	3	variable	external

An implication table is built. The entry corresponding to the variable 'Body covering' is given below.

		Heart Chambers	Body Temperature	Fertilization
hair or feathers	--->	4	regulated	internal
cornified skin	--->	imp. 4	variable	internal
moist skin	--->	3	variable	internal or external

Note that implications for hair and feathers would be combined regardless of any user action since there is perfect overlap in the right-hand sides of these implications.

Computing the Inter-cluster difference and average simplicity

for the 'Body covering' entry yields values of 2/3 and 11/12, respectively. Given Inter-cluster difference and simplicity are of equal importance in the evaluation of table entries, the entry for 'Body covering' is selected as that entry from which divisive rules are constructed.[4]

5. Related Work

The area of conceptual clustering is new. A taxonomy, first suggested in discussion by Michalski and Stepp [6] may be useful in guiding comparative experimental study of various conceptual clustering algorithms. The taxonomy is an analog of one which is frequently reflected in the literature on numerical taxonomy [1].

The taxonomy distinguishes those conceptual clustering algorithms which insist on mutually-disjoint partitions and those which allow for overlapping clusters. The latter are termed Clumping algorithms and examples of these algorithms can be abstracted from the IPP and UNIMEM projects by Lebowitz [2, 3] and found in Stepp [10]. Clusters produced by RUMMAGE are, however, mutually-disjoint and we will concentrate on comparisons with conceptual clustering algorithms of this type.

Conceptual clustering algorithms which insist on mutually-disjoint partitions can be further divided into

---

[4]Although entries for variables 'Heart' and 'Fertilization' have better inter-cluster difference measures, ie. 5/6, their average simplicity measures are quite bad.

Partitioning and Hierarchical techniques. Outwardly, the major difference between these techniques is that hierarchical techniques form decision trees over an object set, whereas partitioning technique result in a 'flat' partition of an object set. The Clustering Module of the CLUSTER/2 program by Michalski and Stepp can be viewed as a conceptual partitioning technique. The Hierarchy-building module of CLUSTER/2 and RUMMAGE are each conceptual hierarchical techniques.

A more fundamental difference between partitioning techniques and hierarchical techniques of the variety found in numerical taxonomy, and reflected in the above examples of conceptual partitioning and hierarchical techniques, is that partitioning techniques explicitly search for an optimal clustering of objects. Given N clusters are desired over an object set, a partitioning technique will search for the optimal set of N clusters. This is the approach taken by the Clustering module of CLUSTER/2, which appears to run in exponential time, that is time proportional to M^N, where N is the number of clusters desired, and M is a function of the number of objects and number of variables over which objects are defined. In contrast, hierarchical techniques depend on 'good' clusterings arising from a series of 'local' choices. For example, the Hierarchy-building module of CLUSTER/2 seeks only to find the optimal division of objects at individual nodes, by calling the Clustering module of CLUSTER/2 for small values of N (ie. where N is now the maximum branching factor of the decision tree being constructed). The

Hierarchy-building module appears to run in polynomial time of degree L, where L is the maximum branching factor of the constructed decision tree.

RUMMAGE, like the Hierarchy-building module of CLUSTER/2 (henceforth, simply CLUSTER/2) depends on 'good' clusterings arising from a series of local choices. RUMMAGE is, however, a computationally cheaper algorithm than CLUSTER/2. The reason for this is two-fold. First, RUMMAGE constructs decision trees which at each node divide an object set by values along a single variable (ie. RUMMAGE constructs **monothetic** classifications), whereas CLUSTER/2 divides nodes based on values along many variables (ie. CLUSTER/2 constructs **polythetic** classifications). The form of rules constructed by CLUSTER/2 subsumes those of RUMMAGE (ie. at each node CLUSTER/2 appears to search a larger space of possible rules). Further, at each node, CLUSTER/2 seeks to find the set of optimal classification rules. In contrast, RUMMAGE conducts a search analogous to **satificing** search as described by Simon [8]. That is, RUMMAGE constructs a set of satisfactory rules for node division, (ie. satisfactory, in that each considered rule has some user controlled minimal degree of inter-cluster difference), and chooses the best among this satisfactory set.

The methods used by RUMMAGE which contribute to its computational efficiency are also responsible for a probable reduced 'robustness' in discovering good clusterings in certain

domains. This is, perhaps, a weakness of any monothetic technique which might be forwarded. Although RUMMAGE is capable of discovering good clusterings implied by single variable values, it is, in general, incapable of discovering good clusterings in a domain in which only a conjunction of several values imply good clusterings. Discussions by Mervis and Rosch [4], however, suggest that in many real-world domains, correlations among features may be such that monothetic techniques would be sufficient for discovery of good object groupings.

6. Future Work

At present RUMMAGE can only handle objects described over **nominal** variables. RUMMAGE is currently undergoing modification so that **structured** (ie. structured by a generalization hierarchy) and **linear** (ie. quantitative) variable types can be used in object description.

The notion of **predictive** features, as described by Lebowitz [2] provides the basis of longer range extensions to RUMMAGE. Lebowitz defines a feature as predictive if its presence in an object can be used to predict the presence of other features (which are termed **predictable**) of the object with a high degree of 'confidence'. A **pragmatic** concept is a concept for which the presence of one of the concept's predictive features in an object, implies with high confidence the presence of all predictable features of the concept. Given the mutually-disjoint nature of object clusters formed by RUMMAGE, notions of

predictive features and pragmatic concepts can be easily formalized in terms of conditional probabilities. With this formalization, RUMMAGE will be modified to form pragmatic object clusters. A related extension is to treat objects not covered by any cluster (ie. exceptions introduced by the formation of pragmatic clusters) as 'fuzzy' members [9] of some number of formed clusters. This treatment of exceptions seems more natural than those reported in earlier systems.[5]

7. Concluding Remarks

We have introduced a conceptual clustering algorithm, RUMMAGE, which forms hierarchical monothetic classifications. The methods employed by RUMMAGE make it a computationally efficient alternative to previously reported conceptual clustering techniques, with a probable associated reduction in the algorithms ability to discover good clusterings in some domains.

---

[5]Lebowitz [2, 3] appears to 'throw out' exceptions once they serve to reduce confidence in appropriate predictive features. From descriptions of CLUSTER/2 by Michalski and Stepp [6, 7], it is unclear what happens to exceptions.

REFERENCES

[1]
Everitt, B.
Cluster Analysis.
Heinemann Educational Books Ltd., 1974.

[2]
Lebowitz, M.
Correcting Erroneous Generalizations.
Cognition and Brain Theory:367-381, 1982.

[3]
Lebowitz, M.
Generalization From Natural Language Text.
Cognitive Science:1-40, 1983.

[4]
Mervis, C. and Rosch, E.
Categorization of Natural Objects.
Annual Review of Psychology:89-115, 1981.

[5]
Michalski, R. and Stepp, R.
Revealing Conceptual Structure in Data by Inductive Inference.
Ellis Horwood Limited, 1982.

[6]
Michalski, R. and Stepp, R.
Automated Construction of Classifications: Conceptual Clustering Versus Numerical Taxonomy.
IEEE Transactions on Pattern Analysis and Machine Intelligence:396-409, 1983.

[7]
Michalski, R. and Stepp, R.
Learning From Observation: Conceptual Clustering.
Tioga Publishing Company, 1983.

[8]
Simon, H.
Sciences of the Artificial.
The MIT Press, 1969.

[9]
Smith, E. and Medin, D.
Catagories and Concepts.
Harvard University Press, 1981.

[10]
Stepp, R.
Learning Without Negative Examples via Variable-Valued Logic Characterizations: The Uniclass Inductive Program AQ7UNI.
Technical Report 79-982, University of Illinois, Urbana-Champaign, 1979.