

A LOGIC-BASED APPROACH TO
OPTIMAL CLASSIFICATION INTO A
LARGE NUMBER OF CLASSES

by

Ryszard S. Michalski

An Internal Report, August 1974.

A LOGIC-BASED APPROACH TO OPTIMAL CLASSIFICATION
INTO A LARGE NUMBER OF CLASSES

R. S. Michalski

Department of Computer Science
University of Illinois
Urbana, Illinois

Internal Report

August 1974

ABSTRACT

The paper is addressed to the problems of designing very efficient classification systems, which are based on the application of variable-valued logic¹⁻⁴. High efficiency of classification is achieved by using the 'simplest' (in a well-defined sense) descriptions of classes of objects. These descriptions are in the form of expressions of the variable-valued logic system VL_1 , and are inferred from examples of objects of known class membership, in the process of inductive inference.

The methodology described in this paper can be especially useful for solving classification problems which:

- involve a very large number of classes, (because the class descriptions generated in this approach are computationally very efficient),
- are intrinsically non-linear, or 'multimodal',
- involve nominal variables (i.e. variables whose values have no meaningful arithmetic relationships because they are labels of independent objects),
- assume that only a small number of representative learning samples is available,
- make it desirable that classification procedures are easily comprehended (this is achieved because the classification procedure resembles the human inferential process).

A LOGISTIC APPROACH TO OPTIMAL CLASSIFICATION
INTO A LARGE NUMBER OF CLASSES

R. S. Michalski
Department of Computer Science
University of Illinois
Urbana, Illinois 61801 U.S.A.

1. INTRODUCTION

Any classification system in order to classify objects into a predetermined set of classes has to know some kind of description of the classes and, also, a procedure for matching these class descriptions with the descriptions of the individual objects. A great majority of works done on classification problems deal with a relatively small number of classes. Therefore, problems of simplicity of class descriptions and the consequent computational efficiency of matching procedures, have not received much attention.

In the case when the number of classes is very large, these problems become important, and a designer of a classification system has to be concerned with not only the adequateness of class descriptions (to achieve high reliability of classification) but also with their computational efficiency as well. By the latter we mean that manipulating, modifying, and storing class descriptions and, as well, using them in the processes of determining class memberships, should require few and simple operations and little memory. This efficiency is directly related to the number of descriptors* used in the class descriptions and to the kind and number of operations involved in matching the class descriptions with the object descriptions.

*By a descriptor we mean a variable, which corresponds to a multi-valued r -ary relation ($r = 1, 2, 3, \dots$) used to characterize objects (or measured on objects). If $r = 1$, the relation is called a feature or a property, and its values are called attributes.

The classification problems with large number of classes arise, for example, in developing a computer-aided consulting system for the first stages of the medical diagnostic process (so-called differential diagnosis). The problem here is to quickly screen out nonpertinent diseases from a very large number of potential diseases, and arrive at a set of most probable diseases, which are subject to further consideration in latter stages of the diagnostic process.

Another example is the situation when, even if the pertinent class of diseases is known, it still may be very large. For example, if this class includes eye diseases, then there can be approximately 1000 such diseases involving about 2000 symptoms.

In this paper we are addressing ourselves to an approach to designing classification systems, which is oriented toward creating the simplest (in a well-defined sense) and computationally very efficient descriptions of classes of objects. This approach stems from the developments of non-classical logic, in particular, the so-called variable-valued logic.¹⁻⁴

It should be noted, that logic generally, seems to be a proper source of theoretical tools for solving problems related to simplicity and efficiency of descriptions. This is so because logic has always been concerned with studying the simplest operations used by humans to process information, and the simplest relationships between objects, namely deterministic relationships in the form of logical consequence.

In this paper we will consider classification methods which are based on a particular variable-valued logic system, called* VL_1 .^{3,4}

* For the lack of space, we will not give here the definition of the VL_1 system. A Reader, in order to understand theoretical background and all the details of this paper, should be acquainted with at least one of the papers¹⁻⁴ (the most complete definition of VL_1 is given in paper⁴). We will give here, however, examples of the VL_1 formulas and their interpretation and, therefore, it is possible to understand the contents of this paper to a satisfactory degree even without knowledge of the above papers.

These methods, which we will call VL₁-based methods, have a number of properties making them significantly different from the established pattern recognition methods, and very attractive from the viewpoint of the efficiency of class descriptions.

The most important properties of these methods can be summarized as follows:

1. The methods have the ability to describe object classes by independent sets of descriptors, selected from an assumed set of available descriptors, as those most appropriate (simplest or 'cheapest') for describing individual classes. This is done by an inductive process performed by a computer program³ AQUAL/1, which learns the class descriptions from samples of object descriptions with known class membership.
2. The methods have capability of using descriptors which are (multivalued) variables measured on nominal or ordinal scales (nominal* or ordinal variables, respectively).

This is a unique feature of VL₁-based methods because the conventional decision space methods require** that variables be measured on ratio or interval scale (the ratio or interval variables, respectively). (The reason for this requirement is the fact that the conventional methods involve arithmetic operations on variables.)

* Also called cartesian or linguistic variables. Nominal variables take values that are labels ('numerical names') of independent objects. Therefore arithmetic relationships between these values have no meaning, and, consequently nominal variables should not be used in arithmetic operations.

** Except for multidimensional scaling methods which handle ordinal variables⁵, and situations when nominal variables are just binary variables.

3. The methods seem to be especially useful for solving intrinsically non-linear classification problems, or problems in which each class includes a number of different object clusters (e.g., when each class corresponds to a set of not-connected regions in a decision space).
4. The methods produce descriptions of classes in a form of expressions involving predominantly logical or set-theoretic operations. The evaluation of these expressions (for determining class memberships) and, also, any manipulation of them (e.g., modification) is very simple and can be done very efficiently from the computational viewpoint. They are also very well suited for parallel or parallel-sequential evaluation, and, therefore, even very large number of such expressions could be processed very quickly.
5. The above classification expressions can be very easily interpreted by humans. This is important in the situations when a human user may be interested in understanding the reasons behind the classification decisions made by a system (such a situation exists, e.g., in computer-aided medical diagnosis).

Let us make a few further comments with regard to property 1.

In conventional decision (feature space) methods, class descriptions are usually in the form of probabilistic or algebraic expressions which depend on the same variables (features) for every class. It means in geometrical interpretation, that the classes correspond to certain n -dimensional domains in the n -dimensional decision space. Such methods, which

we will call constant decision space methods, do not have mechanisms for selecting, from the original set of variables, the subsets of variables which are most suitable for each class. Also, to decide about a class membership, knowledge of values of all variables describing an object is required, no matter to which class the object belongs.

The above properties are significant restrictions of the conventional methods. It is easy to notice that, e.g., humans in describing (and recognizing) objects use class descriptions involving variables (descriptors) which usually are most suitable for describing each individual class. Consequently, each class may be described by a different set of descriptors. Thus, as we will say, they use a variable decision space method. For example, in describing a letter 'H', one might say that it 'consists of 2 parallel vertical bars linked in the middle by a horizontal short bar'. But when describing a letter 'O', one might say that it is 'an ellipse elongated vertically'. The only common descriptor in both descriptions is the concept 'vertical'.

The need for developing variable decision space methods is becoming recognized. An example of such a method is described by Watanabe⁶. A first, to the author's knowledge, implemented recognition system which used independent sets of descriptors (determined in a learning process) for describing individual object classes, was described in paper⁷.

In this paper we will describe two V_{L_1} -based methods for developing optimal (more precisely, near-optimal) classification rules, according to an assumed cost (or optimality) functional, and will illustrate the methods by an example. Before doing it, in the first section of the paper, we will define some basic concepts which will set the theoretical foundations for our treatment of classification problems.

2. DEFINITION OF THE CLASSIFICATION RULE AND OF A DISJOINT REPRESENTATION OF OBJECT CLASSES.

To specify a classification problem the following triple of sets has to be defined:

$$\langle \mathcal{O}, R, K \rangle \quad (1)$$

where:

\mathcal{O} is a non-empty set of physical or abstract objects, called the universe of objects,

R is a finite non-empty (f.n.) set called the universe of representations of objects from \mathcal{O} ,

K is a f.n. set of subsets of \mathcal{O} , called the family of object classes:

$$K = \{K_1, K_2, \dots, K_m\} \quad (2)$$

$K_j \in 2^{\mathcal{O}}$ are called object classes and are specified as

$$K_j = \{o_{j1}, o_{j2}, \dots\}, o_{jk} \in \mathcal{O}, k = 0, 1, 2, \dots \quad (3)$$

Defining notation:

$$K = \{K_1, K_2, \dots, K_m\}^{\cup} \triangleq \bigcup_{j=1}^m K_j \quad (4)$$

(K to the power union) we will assume that

$$K^{\cup} = \mathcal{O} \quad (5)$$

Thus, for each object $o \in \mathcal{O}$, there exists at least one object class containing it. Index j in class K_j is called the numerical name of K_j . The set J indexing classes K_j :

$$J = \{j | K_j \in K\} = \{1, 2, \dots, m\} \quad (6)$$

is called the family of numerical class names.

In general, the index j can assume not only numerical values but non-numerical names or decisions uniquely assigned to classes K_j , and then J is called the family of class names (or decisions).

For illustration, two examples of triples $\langle \emptyset, R, K \rangle$ are given in table below.

No.	\emptyset	R	K
1)	<ul style="list-style-type: none"> • A set of people 	<ul style="list-style-type: none"> • A set of pictures or voice records of these people • A set of their fingerprints • A set of data describing their medical history, physical exam, and lab. tests • A set-theoretical sum of any two or more of the above sets 	<ul style="list-style-type: none"> • A set of groups of people, each group including people with the same disease • A set of groups of people, each group including people of the same nationality
2)	<ul style="list-style-type: none"> • A set of soils 	<ul style="list-style-type: none"> • A set of vectors, each describing one soil sample. Components of vectors specify various properties of soil, such as acidity level, nitrogen content, amount of sand, etc. 	<ul style="list-style-type: none"> • A set of groups of soils, each group including soils where the same crop should be planted

Let:

τ denote a relation between \emptyset and J , called the reference relation:*

$$\tau: \emptyset \text{ -- } J \quad (7)$$

ρ denote a relation between \emptyset and R , called the representation relation:

$$\rho: \emptyset \text{ -- } R \quad (8)$$

κ denote a relation between R and J , called the classification relation:

$$\kappa: R \text{ -- } J \quad (9)$$

* By $l: S_1 \text{ -- } S_2$, where S_1 and S_2 are sets, is meant that l is a relation between S_1 and S_2 (that is, $l = \langle S_1, S_2, G_l \rangle$, $G_l \subseteq S_1 \times S_2$). And by $l(s)$, $s \in S_1$, is meant the set of elements from S_2 which are related by l to s .

In the present paper we will restrict ourselves to the case where K consists of disjoint sets and τ and κ are functions:

$$\tau: O \rightarrow J \quad (10)$$

$$\kappa: R \rightarrow J \quad (11)$$

(that is, there exists only one class related to any object and any representation). Figure 1 presents schematically relations τ , ρ , and κ .

If l is a relation between sets S_1 and S_2 :

$$l: S_1 \dashrightarrow S_2 \quad (12)$$

then by $Ex(l)$ we denote an expression for l , which is a formula consisting of variables (whose values depend on elements of S_1), and different operations (e.g., logical, arithmetic, control), such that for any $s \in S_1$, $Ex(l)$ computes elements of S_2 related by relation l to element s .

Suppose that for semantical restrictions (e.g., because a mathematical formula cannot deal directly with physical objects) or practical reasons (e.g., because sets O and/or R are too large) it is not feasible to specify relations τ and ρ completely and to determine expressions for them. Suppose, however, that τ and ρ are specified for some subsets $O \subseteq O$ and $R \subseteq R$ such that O is the union of non-empty disjoint sets O_1, O_2, \dots, O_m :

$$O_j \subseteq \{o | o \in K_j\}, j = 1, 2, \dots, m \quad (13)$$

and R is the union of non-empty disjoint sets R_1, R_2, \dots, R_m :

$$R_j \subseteq \{r | r \in \rho(o), o \in O_j\} \quad (14)$$

such that for every $o \in O_j$, there is in R_j at least one $r \in \rho(o)$.

Definition 1.

Set R , as described above, is called a disjoint representation of the set O . If it is not required that sets R_j must be disjoint (for a certain

R there may not exist a disjoint representation of \mathcal{C}), then R is called a non-disjoint representation of \mathcal{C} .

A non-disjoint representation does not make possible the classification of objects without error. In such a case the following approaches can be taken:

1. Evaluate class-conditional probability densities $f_j(r)$ of an object with representation r , being from class K_j , $j = 1, 2, \dots, m$ and then construct decision rules which, e.g., give the minimum expected error in classification based on representations $r \in R$ (such methods are studied in statistical pattern recognition).
2. Extend the universe of representations R , or seek a new R , such that it will provide a disjoint representation. (A general tendency here should be to seek an R such that sets R_j are not only disjoint but form in R 'clusters' -- one or more per class.) Such a representation R is called compact or clustered.

In this paper we are concerned only with the second approach.

Let τ and ρ denote restrictions of τ and ρ :

$$\tau: \mathcal{C} \rightarrow J \quad (15)$$

$$\rho: \mathcal{C} \rightarrow R \quad (16)$$

Let κ denote a function:

$$\kappa: R \rightarrow J \quad (17)$$

such that the composite relation $\rho \circ \kappa$ is equal to τ :

$$\rho \circ \kappa = \tau \quad (18)$$

Note that in order to satisfy (18), function κ has to have appropriate values only for elements of $R \subseteq \mathbf{R}$, and, therefore, there can be many functions of type (17) which satisfy (18). Each of them has the property that $\rho \circ \kappa$ is equal to τ for objects $o \in \mathcal{O}$, and for objects $o \notin \mathcal{O}$ may or may not be equal to τ .

Definition 2.

An expression $Ex(\kappa)$ is called a classification rule for the set \mathcal{O} based on R into J , or briefly, a classification rule $C(\mathcal{O}, R, J)$.

Some of the types of problems which arise are:

1. How, given τ and ρ , to determine a classification rule $C(\mathcal{O}, R, J)$ which
 - (a) involves only operations from some specified set,
 - (b) is minimal under an assumed cost functional, or, generally, satisfies certain criteria.
2. How, given a classification rule $C(\mathcal{O}, R, J)$, to estimate its performance for objects $o \notin \mathcal{O}$ and/or representations $r \notin R$ (assuming that τ can be specified for some additional objects).
3. How, given τ and ρ , to determine a set of operations, such that classification rules involving all or some of these operations can be made very 'simple' (in a specified sense).
4. How, given \mathcal{O} and τ , to specify a universe R which will provide a disjoint (or clustered) representation and will permit construction of very 'simple' classification rules.

In the present paper we will describe two methods for solving problems of type 1, where allowed operations are those which are defined in the system VL_1 . We will also discuss and illustrate by an example a problem of designing a disjoint representation of objects (i.e. a problem of type 4).

3. VL_1 -BASED CLASSIFICATION METHODS

3.1 Event Space as the Universe of Representations

By VL_1 -based classification methods we mean methods of using the VL_1 system for classification purposes. These methods assume that the universe of object representations R is a set of vectors whose components take values from certain finite sets D_i , $i = 1, 2, \dots, n$. Such a universe of representations is called an event space and is denoted as $E(d_1, d_2, \dots, d_n)$ or, briefly, E , where d_i , $i = 1, 2, \dots, n$, are the cardinalities of D_i . Thus we have:

$$E(d_1, d_2, \dots, d_n) = D_1 \times D_2 \times \dots \times D_n \quad (19)$$

where* $d_i = c(D_i)$.

Elements of an event space are called events. Sets D_i , $i = 1, 2, \dots, n$, are interpreted as domains of certain variables. These variables are, generally, identified by names and no order among the variables is implied. To be specific, however, we will assume that these variables are x_1, x_2, \dots, x_n , and their domains are D_1, D_2, \dots, D_n , respectively. Variables x_i , $i = 1, 2, \dots, n$, correspond to certain descriptors (features, properties, relations, or domains of relations) which are used to characterize objects (e.g., various measurements of objects). Elements of D_i , i.e. values of variables x_i , can be numbers (e.g., results of measurements) or, in the most general case, names of any objects.

* $c(S)$, where S is a set, denotes the cardinality of S .

In the AQVAL/1 implementation³ of the system VL₁, (to which we will refer in this paper) it is assumed that domains D_i are sets of non-negative integers:

$$D_i = \{0, 1, 2, \dots, \alpha_i\} \quad (20)$$

where $\alpha_i = d_i - 1$.

If the original domains of variables are not such sets of non-negative integers, then they should be mapped into such sets. This mapping can be accomplished in the following way.

If an original domain of a variable is a finite set, then the elements of the domain are ordered according to some desired semantic or syntactic property, and are assigned positions 0, 1, 2, ..., α_i . These positions will then be taken as values of variables (variables with domains in the form (20) will from now on be called AQVAL/1-variables). By 'ordering according to a semantic property' we mean an ordering based on the meaning of values of variables. For example, consider a variable 'height of a person'. Its values can be measurements of a person's height expressed in centimeters, or in words, such as 'tall, short, average, very tall', depending on the degree of preciseness required in a given classification problem. If the latter case is assumed, then an ordering of the domain according to a semantic property could be 'short, average, tall, very tall', reflecting increasing height described by these adjectives. By an 'ordering according to a syntactic property' we mean, e.g., an alphabetic order. A syntactic ordering is used when a domain is a set of names of independent objects, e.g., a set of first names {Helen, Barbara, Hanna, Renata} (a variable with this domain is a nominal variable).

If a domain is an interval of a real line (i.e., in the case of a continuous variable), it is quantized into discrete units, which are assigned positions $0, 1, 2, \dots, \alpha$, taken as values of an AQVAL/1-variable (the problem of how to quantize continuous variables goes beyond the scope of this paper).

3.2. VL₁ System

A full definition of the variable-valued logic system VL₁ was given in papers^{3,4}. Here we will restrict ourselves to only some basic informal description of the system and illustrate it by examples.

The VL₁ logic is an extension of a many-valued logic system, in the sense that formulas of the system and all the variables in the formulas are allowed to range over independent finite domains. The domains (whose elements can be interpreted on the grounds of logic as 'truth values') are determined based on semantic or problem oriented considerations (i.e., are semantic- or problem-dependent, as opposed to the traditional treatment of the concept of 'truth' in (binary or multi-valued) logic, in which all the variables and formulas are assumed to range over the same domain (of truth-values)).

The well-formed formulas in VL₁ (VL₁ formulas) are interpreted as expressions of a function (called a VL function):

$$f: D_1 \times D_2 \times \dots \times D_n \rightarrow D \quad (21)$$

where

$$D_i = \{0, 1, 2, \dots, \alpha_i\}, \quad i = 1, 2, 3, \dots, n \quad (22)$$

$$D = \{0, 1, 2, \dots, \alpha\} \quad (23)$$

or, equivalently,

$$f: E(d_1, d_2, \dots, d_n) \rightarrow D \quad (24)$$

where $d_i = \bar{x}_i + 1$.

The system is complete, that is for every function (21), there exists at least one VL_1 formula which is an expression of this function. Following is a simple example of a VL_1 formula and its interpretation.

Example 1.

$$3[x_2 \geq 2][x_4 = 2:5, 7][x_5 \neq 0] \vee 2[x_3 = 0, 3] \vee 1[x_1 \leq 3][x_2 = 3] \quad (25)$$

The forms in brackets are called selectors. They represent conditions which a value of the variable within a selector must fulfill in order to satisfy the selector. If a selector is satisfied, then it is assigned the highest value in the domain D , that is \bar{x} , otherwise value 0. Constants outside the selectors are elements of domain D . Concatenations of selectors and a constant (from D) are interpreted as logical products (conjunctions) and are called terms. Logical products are evaluated as the minimum of values of their components (the value of a constant is the constant itself). Terms are linked by the symbol ' \vee ' denoting a logical sum (disjunction). The logical sum is evaluated as the maximum of all the values of the terms.

A VL_1 formula in the form of a logical sum of terms (e.g. (25)) is called a disjunctive simple VL_1 formula or, simply, a DVL₁ formula.

Assuming that the domains of variables x_1, x_2, x_3, x_4 and x_5 are respectively:

$$\begin{aligned} D_1 &= \{0, 1, 2, 3\} & D_2 &= \{0, 1, 2, 3, 4\} \\ D_3 &= \{0, 1, 2, 3\} & D_4 &= \{0, 1, 2, 3, 4, 5\} \\ D_5 &= \{0, 1, 2\} \end{aligned}$$

and the domain of values of the formula is $D = \{0, 1, 2, 3\}$, the formula (25) is interpreted as an expression of a function:

$$f: D_1 \times D_2 \times D_3 \times D_4 \times D_5 \rightarrow D \quad (26)$$

The formula (25) is assigned value 3 (has value 3), if the value of variable x_2 is greater or equal 2, the value of variable x_4 is between 2 and 5, inclusively, or is 7, and the value of variable x_5 is not 0. (Note that the values of variable x_1 and x_3 have no affect on this outcome.)

The formula has value 2, if the above compound condition does not hold (i.e., if one or more selectors is not satisfied), and the value of variable x_3 is 0 or 3.

If both of the above compound conditions do not hold, and, if the value of variable x_1 is smaller than or equal to 3 and the value of variable x_2 is 3, then the formula has value 1.

If none of the above compound conditions hold, the formula has value 0.

Example 2.

$$2[x_2+x_3=1,2] \vee 1[x_1=0,4] \quad (27)$$

This formula illustrates how a symmetry with regard to some or all variables can be expressed in the VL_1 system and, also, illustrates a so-called exception operation (denoted by \vee). The formula is interpreted as follows: it has value 2 if the values of x_2 and x_3 arithmetically sum to 1 or 2, except when variable x_1 has value 0 or 4. In the latter case, the formula is assigned value 1. If the above conditions do not hold, the formula has value 0. (The selector in (27), which is concatenated with constant 2, is called a symmetric selector.)

It can be seen from the above examples, that the VL_1 system expresses VL functions by setting various conditions on the values of variables and by linking these conditions by logical connectives.

3.3 AQVAL/1 Program

Let us assume, that in a given classification problem, the universe of representations is an event space E and that the universe of class names is D (20). A classification relation (11) can then be interpreted as a VL function (24). A VL_1 formula which is an expression of this function (a VL_1 expression), becomes a classification rule. For a given VL function, there exist, in general, a very large number of VL_1 expressions of this function. These expressions may involve different numbers of variables and different numbers of operations, and, consequently, different computational costs will be involved in handling, storing, modifying, and evaluating them. Thus the problem arises of how to determine, given a classification relation in the form of a VL function, a VL_1 expression of this relation which will be the most desired (optimal) according to a certain optimality functional.

This optimality functional may reflect, e.g., the computational efficiency of storing and evaluating expressions, or any other desired property, such as the minimal cost associated with evaluating variables in expressions.

A solution to the above problem has been achieved by developing a computer program AQVAL/1*, which, for a given VL function, produces a DVL₁ expression of it, which is near-optimal, or optimal, according to an assumed optimality functional. The program also produces a measure Δ of the maximum possible difference, in the number of terms, between the formula obtained and the formula which has the minimal possible number of terms. Thus, if $\Delta = 0$, the formula obtained has the absolutely minimal number of terms which is possible in a DVL₁ expression of the given VL function.

The description of algorithms used in AQVAL/1, and other information pertinent to this program, are beyond the scope of this paper. A functional description of AQVAL/1 has been given in paper³. Here, we will give only a few basic facts about it, which are necessary to understand how AQVAL/1 can be used for learning object class descriptions, which are the subject of the next section of the paper.

A VL function f can be specified to the AQVAL/1 program in the following ways:

1. By listing sets of events from E , such that each set includes all known events for which a VL function f takes the same value k , $k = 0, 1, 2, \dots, d$. That is, by listing sets:

$$F^k = \{e = (x_1, x_2, \dots, x_n) \mid f(e) = k\}, \quad k = 0, 1, \dots, d. \quad (28)$$

where \dot{x}_i denotes a value of variable x_i .

2. By a DVL₁ expression of the function f .

*The name AQVAL/1 was derived from 'Algorithm AQ Applied for the synthesis of Variable-Valued Logic formulas. The algorithm AQ is a general, very efficient algorithm⁸⁻¹¹ for solving covering problems (in particular for minimization of multi-variable logic functions) which was used in AQVAL/1.

In case 2, AQVAL/ will try to optimize, if possible, the given expression according to the user-specified optimality functional.

An optimality functional is specified in AQVAL/1 in the form:

$$A = \langle a\text{-list}, \tau\text{-list} \rangle \quad (29)$$

where

a-list, called the attribute (or criteria) list, is a vector $a = (a_1, a_2, \dots, a_l)$, where the a_i denote single- or many-valued attributes used to characterize DVL₁ formulas,

τ -list, called the tolerance list, is a vector $\tau = (\tau_1, \tau_2, \dots, \tau_l)$, where $0 \leq \tau_i \leq 1$, $i = 1, 2, \dots, l$, and the τ_i are called tolerances for attributes a_i .

A DVL₁ formula V is said to be an optimal DVL₁ expression for f under functional A iff:

$$A(V) \preceq A(V_j) \quad (30)$$

where

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

$$A(V_j) = (a_1(V_j), a_2(V_j), \dots, a_l(V_j))$$

$a_i(V)$, $a_i(V_j)$ denote the value of the attribute a_i for formula V and V_j , respectively. V_j , $j = 1, 2, 3, \dots$ denote all irredundant* DVL₁ expressions for f

\preceq denotes a relation, called the lexicographic order with tolerance τ , defined as:

$$A(V) \preceq A(V_j) \text{ if } \begin{cases} a_1(V_j) - a_1(V) > \tau_1 \\ \text{or } 0 \leq a_1(V_j) - a_1(V) \leq \tau_1 \text{ and } a_2(V_j) - a_2(V) > \tau_2 \\ \text{or } \dots \dots \dots \\ \text{or } \dots \dots \dots \text{ and } a_l(V_j) - a_l(V) \geq \tau_l \end{cases} \quad (31)$$

$$\tau_i = \tau_i \cdot (a_{imax} - a_{imin}), \quad i = 1, 2, \dots, l$$

$$a_{imax} = \max_j \{a_i(V_j)\}, \quad a_{imin} = \min_j \{a_i(V_j)\}$$

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

* A DVL₁ expression of a function f is called irredundant if removing any term or selector from it makes it no longer an expression for f .

A user designs a functional $A = \langle \text{a-list}, \text{r-list} \rangle$ by selecting a set of attributes from the available attributes, placing them in the desired order in the a-list, and setting up the r-list. Presently eight attributes are available in the program. Some of the most important are:

1. $t(V)$ - the number of terms in V ,
2. $s(V)$ - the number of selectors in V ,
3. $z(V)$ - the cost of V specified as $\sum_{i \in I} z(x_i)$, where $z(x_i)$ is the cost, specified in the input data, of determining the value of variable x_i , and I is the set of indices of those variables (among x_1, x_2, \dots, x_n , specified in the input data) which actually appear in the output VL_1 formula.
4. $g(V)$ - the 'degree of generalization' defined as:

$$g(V) = \frac{1}{\xi_0} \sum_{k=1}^{\bar{a}} \sum_{l=1}^{s_k} g(L_{kl})$$

where

$$g(L_{kl}) = \frac{c(L_{kl})}{c(L_{kl} \cap F^k)}$$

L_{kl} -- the set of events which satisfy the l -th term in the sequence of terms in V with the constant k ,

s_k -- number of terms with the constant k

ξ_0 -- total number of terms in V

$F^k = \{e \mid f(e) = k\}$

3.4 Dependent and Independent (VL_1 -Based) Class Descriptions

We will describe here two methods of using AQVAL/1 to determine optimal classification rules, according to an assumed optimality functional:

- method DCD, which produces the so-called dependent class descriptions, and
- method ICD, which produces the so-called independent class descriptions.

3.4.1 Method DCD

Let us assume that the a priori probability that an object (which is to be classified) is from class K_k , $k = 0, 1, 2, \dots, \bar{a}$, is $p(k)$. Let us

assume further, that the probabilities $p(k)$, $k = 0, 1, 2, \dots, \bar{a}$, are ordered:

$$p(\bar{a}) > p(\bar{a}-1) > \dots > p(1) > p(0) \quad (32)$$

and remain constant in time.

Suppose now that $C(k)$, $k = 0, 1, 2, \dots, \bar{a}$, denotes descriptions of classes K_k , respectively. These descriptions $C(k)$ are statements of certain conditions for the values of variables x_i , $i = 1, 2, \dots, n$, (where x_i represent descriptors characterizing objects). If the conditions in a description $C(\dot{k})$, (where \dot{k} denotes a particular value of k), are satisfied, then the object is classified as belonging to class $K_{\dot{k}}$ (or, simply, class \dot{k}). An assumption is made here, that descriptions are evaluated one at a time, and also that if more than one of the descriptions is satisfied, then the object is classified as belonging to the class with the highest a priori probability.

Under these assumptions, to minimize the average number of descriptions which is evaluated before a classification decision is made, the order of evaluation of the descriptions should correspond to the descending order of a priori class probabilities, i.e., first should be evaluated description $C(\bar{a})$, then $C(\bar{a}-1)$, and so on up to $C(0)$.

The fact that descriptions are always evaluated in the same order can be used to simplify these descriptions (due to the 'sieve effect'). Namely, the conditions in a given description $C(\dot{k})$, whose only purpose is to distinguish class \dot{k} from classes $\dot{k}+1$, $\dot{k}+2$, \dots , \bar{a} can be removed from the description $C(\dot{k})$.

To achieve a classification rule which takes advantage of the above assumptions, the AQVAL/1 program should be used in the following way. Elements of the domain D should be assigned to classes k in such a way that

d represents the class with the highest a priori class probability, $d-1$, the next highest, etc., up to 0, which represents the class with the lowest a priori class probability. Learning samples (in the form of events $e \in E$) for classes k , $k = 0, 1, \dots, d$, should be specified as sets F^k . AQVAL/1 will produce a DVL_1 expression for the function

$$f: E \rightarrow D \quad (33)$$

defined by sets F^k (according to equation $F^k = \{e | f(e) = k\}$, $k = 0, 1, \dots, d$). The obtained expression will be quasi-optimal (or quasi-minimal*) according to a user specified functional A . By 'quasi-optimal' we mean optimal or approximately-optimal.

In such an obtained formula, the description of a class k , $k = 1, 2, \dots, d$, is represented by the logical sum of terms with constant k . There is, obviously, no term with constant 0, thus the class 0 is not represented explicitly in the formula, but implicitly, i.e., is assigned to those events which do not satisfy any term in the formula. These class descriptions are interrelated in the sense that satisfying a term with constant k is a necessary, but not sufficient condition for classifying an object to class k . The sufficient condition requires also that none of the terms with a constant $k > k$ is satisfied. The above explains the name DCD (dependent class descriptions) given to the method described.

3.4.2 Method ICD

This method produces independent class descriptions. It can be used when:

*The adjective 'optimal' is, generally, preferable to 'minimal', because in certain situations a user may require that a formula satisfies a certain condition (e.g., does not involve, if possible, certain variable), rather than that minimizes some criteria.

a) class descriptions are to be evaluated in parallel, or
 b) they are to be evaluated sequentially (as in method DCD),
 but a priori class probabilities vary for each time an object is to be classified. The latter situation exists, e.g., when the classification system receives, together with the representation (of the object to be classified), also a preliminary hypothesis about the class membership of the object.* In this situation, the classification system should first of all test the given hypothesis. Therefore, class descriptions should be independent of each other, which means that satisfying a description of class k would become the sufficient condition of classifying an object to class k .

Such independent descriptions are created by describing each class by a single VL_1 formula which can take only two values 0 or 1, that is, which is an expression of a function

$$f: E \rightarrow \{0,1\} \quad (34)$$

If the formula assumes value 1, then the class description associated with this formula is considered to be satisfied, if 0 -- not satisfied.

A set of independent class descriptions is obtained by running AQVAL/1 in the so-called 'IC' or 'DC' modes³. In the 'IC' mode AQVAL/1 produces a set of VL_1 formulas, each being an independent generalization of the learning samples of individual classes. Therefore, some formulas describing different classes may (potentially) intersect, i.e., may be satisfied by the same events (however, they will not be simultaneously satisfied by any events from the learning sets). In the 'DC' mode, all formulas obtained will be pairwise disjoint.

* An example of such a hypothesis is an 'admitting diagnosis' given to a patient admitted to a hospital.

The penalty for having independent class descriptions is that they will usually be more 'complex' than dependent class descriptions (i.e., will have more terms and/or more selectors).

In the following section we will illustrate both methods, DCD and ICD, by an example.

4. AN EXAMPLE OF DEVELOPING A VL_1 CLASSIFICATION RULE

Figure 2 presents 14 groups of 'animals*' (or 'shady characters'). Let us assume that each group includes all possible examples of 'animals' of the same one species. Suppose now that the problem is to design a very efficient classification system, which for a given 'animal' from Figure 2, determines the species to which it belongs.

A. The first step in solving such a problem, by the VL_1 -based approach, is to design an event space which serves as the universe of representations of all objects (i.e., in this case 'animals').

The problems of designing such a universe of representations go beyond the scope of this paper. We will restrict ourselves to only a few general superficial remarks. The step A generally consists of the following substeps:

1. Designing a basic set of descriptors.

Here the designer should use all the available knowledge of the problem to determine what kind of descriptors may be 'relevant to the given problem'. In the basic set of descriptors, any relevant descriptor can be included which can:

- discriminate between at least two classes of objects,
- be measured by available devices (or by known procedures).

The obtained set of descriptors should provide a disjoint representation of objects (recall section 2).

2. Minimizing the domains of descriptors.

Domains of descriptors should be minimized, that is, should include as small as possible a number of elements, preserving the

*The 'animals' were selected (after slight modification) from problem cards in the book 'Altitude Games and Problems', by Educational Development Center, Inc., published by the Webster Division of McGraw-Hill Book Co., 1968.

disjoint object representation. Ranges of variability of continuous descriptors should be quantized into the smallest acceptable number of discrete units (such quantization is again a separate problem, beyond of the scope of this paper).

3. Mapping the domains obtained into AQVAL/1 domains, i.e. sets

$$D_i = \{0, 1, \dots, d_i\}$$

and accepting the D_i as domains of variables x_i , $i = 1, 2, \dots, n$.







(Substeps 2 and 3 can be performed with the aid of a computer program called 'QUANT'.)

For the example under consideration, we have designed the following

set of descriptors and their domains:

x_1 -- number of black circles on the body $D_1 = \{0, 1, 2\}$, where:
 0 -- denotes no black circles
 1 -- 1
 2 -- 2 or more

x_3 -- number of crossmarks on tails $D_3 = \{0, 1, 2\}$, where:
 0 -- no crossmarks
 1 -- one or two
 2 -- three

x_5 -- type of body texture $D_5 = \{0, 1, 2, 3, 4, 5, 6\}$, where:
 0 -- blank
 1 -- 
 2 -- 
 3 -- 
 4 -- 
 5 -- 
 6 -- 

x_7 -- number of empty squares on the body $D_7 = \{0, 1\}$, where:
 0 -- no squares
 1 -- one or more

x_9 -- type of tail $D_9 = \{0, 1, 2\}$, where:
 0 -- no tail or more than one
 1 -- straight tail
 2 -- spring tail

x_2 -- number of tails $D_2 = \{0, 1\}$, where:
 0 -- no tails
 1 -- one or more

x_4 -- number of easily distinguished extremities $D_4 = \{0, 1\}$, where:
 0 -- none or one extremity
 1 -- two or more

x_6 -- number of empty circles on the body $D_6 = \{0, 1, 2\}$, where:
 0 -- none or one circle
 1 -- two
 2 -- three or more

x_8 -- number of empty triangles on the body $D_8 = \{0, 1\}$, where:
 0 -- no triangle
 1 -- one or more

x_{10} -- shape of the body $D_{10} = \{0, 1, 2, 3\}$
 0 -- irregular (other than specified as 1, 2 or 3)
 1 -- ellipse
 2 -- circle
 3 -- triangle or square

- x_{11} -- number of sharp or straight angles
 $D_{11} = \{0,1\}$, where:
 0 -- no angles
 1 -- one or more
- x_{12} -- number of 'eyes' (half-black circles)
 $D_{12} = \{0,1\}$, where:
 0 -- no eyes
 1 -- one or more
- x_{13} -- number of black squares on the body
 $D_{13} = \{0,1\}$, where:
 0 -- no black squares
 1 -- one or more

B. The next step is to develop a classification rule with the help of the AQUAL/1 program.

All animals were described in terms of the descriptors x_1, \dots, x_{13} . Descriptions of the members of one species were combined into individual learning sets F^k , $k = 0, 1, \dots, 13$.

Method DCI.

We have assumed the order of classes as it is marked in Figure 2 by numbers $0, 1, \dots, 13$. AQUAL/1 was run twice, for restriction parameters*:

1. $\langle ms, cs \rangle = \langle 5, 2 \rangle$
2. $\langle ms, cs \rangle = \langle 4, 0, 20 \rangle$

The optimality functional was assumed to be $A = \langle t, s, g \rangle$ (recall that t, s, g denote number of terms, number of selectors, and degree of generalization, respectively). In case (1) the formula obtained had 20 terms and 61 selectors. Computation time was 22.6 sec. (on IBM 360/75, AQUAL/1 is written in PL/1). In case (2) the formula obtained had 20 terms and 54 selectors. Computation time was 60.5 sec.

Descriptions $C(k)$, $k = 0, 1, \dots, 13$ of individual classes ($0, 1, \dots, 13$, respectively) based on the VI_1 formula obtained in run (2) are listed below.

$C(0) = [x_6=1][x_8=0][x_{10}=0]$	(5,5)	$\Delta = 0$
$C(1) = [x_3=0][x_4=1][x_6=2][x_{13}=0] \vee$	(1,1)	} $\Delta = 0$
$[x_2=0][x_6=2][x_{13}=0] \vee$	(2,2)	
$[x_1=0][x_2=1][x_6=0][x_9=0][x_{13}=0]$	(2,2)	
$C(2) = [x_6=1][x_{11}=0]$	(5,5)	$\Delta = 0$
$C(3) = [x_1=0][x_6=1,2] \vee$	(4,4)	} $\Delta = 0$
$[x_2=0][x_4=0][x_5=0]$	(3,1)	
$C(4) = [x_1=0][x_5=1][x_8=0][x_9=1]$	(6,6)	$\Delta = 0$

*The smaller the restriction parameters are, the faster AQUAL/1 runs, but the resulting formulas may be further from the optimum (according to functional A).

$C(5) = [x_1=0][x_4=0][x_8=0][x_9=0,1][x_{13}=0]$	(6,6)	$\Delta = 0$
$C(6) = [x_3=0][x_5=1][x_8=0][x_9=1]$	(5,5)	$\Delta = 0$
$C(7) = [x_3=0][x_5=0,1][x_9=1] \vee$ $[x_4=0][x_8=1]$	(4,4) (2,2)	} $\Delta = 0$
$C(8) = [x_9=2][x_{13}=1]$	(6,6)	
$C(9) = [x_2=1][x_3=0][x_5=0,5] \vee$ $[x_1=0][x_2=1]$	(4,4) (4,2)	} $\Delta = 0$
$C(10) = [x_1=2][x_3=2][x_5=1]$	(6,6)	
$C(11) = [x_2=1]$	(7,7)	$\Delta = 0$
$C(12) = [x_5=1,4] \vee$ $[x_8=1] \vee$ $[x_5=0]$	(3,3) (1,1) (1,1)	} $\Delta = 0$

Recall that class descriptions are dependent, i.e., in making a classification decision, the description $C(k)$, $k \in \{0,1,\dots,12\}$, should be evaluated only if none of the descriptions $C(k)$, $k > k$, were satisfied. Class 13 is implied when none of the descriptions $C(k)$, $k = 0,1,\dots,12$, is satisfied. Numbers a and b in the pairs (a,b) on the right-hand side of each term denote, respectively, the total number of 'animals' in a given species 'covered' by the term, and the number of animals 'covered' only by the given term (i.e., not covered by other terms of the description).

Note, that $\Delta = 0$ for all class descriptions, which means that all descriptions have the minimal possible number of terms for this classification problem.

Let us interpret some of the descriptions, e.g., $C(0)$ and $C(8)$.

An 'animal' will be classified as belonging to species 0 (Jexums), if variable x_6 equals 1 (i.e., if the 'animal' has 2 empty circles on the body), variable x_8 equals 0 (i.e., the 'animal' has no triangles on the body) and variable x_{10} equals 0 (i.e., its body is 'irregular'). An animal is classified as belonging to species 8 (Fubbyloofers), if its description does not satisfy any of the descriptions $C(0)$, $C(1)$, ..., $C(7)$, and if variable x_2 equals 2 (i.e., the animal has a 'spring' tail) and if variable x_{13} equals 1 (i.e., the animal

has one black circle on the body). It is easy to see, that in this particular case, the description C(8) is in fact independent of other descriptions, though for other descriptions this may not be true (e.g., description C(11)).

Method ICD.

In this case AQVAL/1 was run in the 'IC' mode, to obtain VL₁ formulas, each discriminating one class from all the others. There were two runs, as before, with different restriction parameters:

1. $\langle ms, cs \rangle = \langle 5, 3 \rangle$
2. $\langle ms, cs \rangle = \langle 200, 100 \rangle$

The optimality functional was $\langle t, s, g \rangle$ (as before).

The formula obtained in run (1) had in toto (for all classes) 31 terms and 121 selectors (the computation time was ≈ 25 sec.). The formula obtained in run (2) had in toto 30 terms and 89 selectors (the computation time was ≈ 90 sec.). Below are listed (independent) descriptions $\hat{C}(k)$, $k = 0, 1, \dots, 13$, of species based on formulas obtained in run (2).

$\hat{C}(0) = [x_6=1][x_8=0][x_{10}=0]$	(5,5)	$\Delta = 0$
$\hat{C}(1) = [x_2=1][x_4=1][x_5=0] \vee$	(1,1)	} $\Delta = 0$
$[x_2=0][x_6=2][x_{13}=0] \vee$	(2,2)	
$[x_1=0][x_2=1][x_6=0][x_9=0][x_{13}=0]$	(2,2)	
$\hat{C}(2) = [x_6=1][x_8=1]$	(5,5)	$\Delta = 0$
$\hat{C}(3) = [x_4=0][x_5=0][x_6=0,2][x_7=0][x_9=0][x_{10}=0][x_{11}=0] \vee$	(3,3)	} $\Delta = 0$
$[x_6=1][x_{10}=3] \vee$	(1,1)	
$[x_2=0][x_{13}=1]$	(2,1)	
$\hat{C}(4) = [x_1=0][x_5=1][x_8=0][x_9=1]$	(6,6)	$\Delta = 0$
$\hat{C}(5) = [x_2=0][x_4=0][x_5=1][x_8=0] \vee$	(2,2)	} $\Delta = 0$
$[x_1=0][x_5=0,6][x_9=1]$	(4,4)	
$\hat{C}(6) = [x_1=1][x_3=0][x_5=1][x_9=1]$	(5,5)	$\Delta = 0$
$\hat{C}(7) = [x_1=1][x_5=0][x_9=1] \vee$	(2,2)	} $\Delta = 0$
$[x_4=0][x_6=0][x_8=1]$	(4,4)	

$\hat{c}(8) = [x_9=2][x_{13}=1]$	(6,6)	$\Delta = 0$
$\hat{c}(9) = [x_5=5][x_{13}=0] \vee$	(1,1)	} $\Delta = 0$
$[x_2=1][x_9=0][x_{13}=1] \vee$	(3,3)	
$[x_9=2][x_{13}=0]$	(2,2)	
$\hat{c}(10) = [x_1=2][x_3=2][x_5=1]$	(6,6)	$\Delta = 0$
$\hat{c}(11) = [x_1=1][x_3=2] \vee$	(2,2)	} $\Delta = 0$
$[x_1=2][x_5=0] \vee$	(2,2)	
$[x_3=1] \vee$	(1,1)	
$[x_1=1][x_2=1][x_9=0]$	(3,2)	
$\hat{c}(12) = [x_2=0][x_4=1][x_5=1,4] \vee$	(3,3)	} $\Delta = 0$
$[x_5=0][x_6=0][x_8=1] \vee$	(1,1)	
$[x_2=0][x_5=0][x_6=0][x_{10}=0][x_{11}=1]$	(1,1)	
$\hat{c}(13) = [x_4=1][x_5=0][x_6=0][x_8=0][x_{11}=0] \vee$	(3,3)	} $\Delta = 0$
$[x_4=1][x_{10}=2] \vee$	(1,1)	
$[x_5=3] \vee$	(1,1)	
$[x_2=2][x_4=1]$	(1,1)	

As we can see, Δ was 0 for all classes, thus the obtained descriptions have the minimum possible number of terms.

All these descriptions are independent, that is, if an 'animal' satisfies description $C(k)$, then it is classified as belonging to class k . It should be noted that for each 'animal' in Figure 2, there will be only one description $C(k)$ which is satisfied (though, in the general case, there exist events, i.e., sequences of descriptor values, which satisfy more than one description).

CONCLUDING REMARKS

The methodology described in this paper is a part of a theoretical basis for an experimental medical inferential diagnostic advice system (MIDAS) being under development at the University of Illinois.

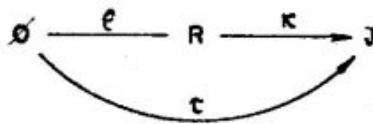
Acknowledgment

The author expresses sincere thanks to Ms. Renata Chimiak and Mr. Shu Kan Yuen for their help in completing the experiment on classification of 'animals'. The author also thanks Mr. Gary Pace for the help in editing the paper, and Mrs. Connie Slovak for her excellent typing of this paper.

References

1. Michalski, R. S., A Variable-Valued Logic System as Applied to Picture Description and Recognition, GRAPHIC LANGUAGES, Proceedings of the IFIP Working Conference on Graphic Languages, pp. 20-47, Vancouver, Canada, May 22-26, 1972.
2. Michalski, R. S., Discovering Classification Rules by the Variable-Valued Logic System VL_1 , Proceedings of the Third International Joint Conference on Artificial Intelligence, Stanford, California, August 20-24, 1973.
3. Michalski, R. S., AQVAL/1--Computer Implementation of a Variable-Valued Logic System VL_1 and Examples of Its Application to Pattern Recognition, Proceedings of the First International Joint Conference on Pattern Recognition, Washington, D.C., October 30-November 1, 1973 pp. 3-17.
4. Michalski, R. S., 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.
5. Milojkovic, D. V., Goetz, A. J. and Tappert, An application of a nonmetric multidimensional scaling method for feature selection in pattern recognition, Proceedings of the First International Joint Conference on Pattern Recognition, Washington, D.C., October 30-November 1, 1973, pp. 356-373.
6. Watanabe, S., Pakvasa, N., Supspace Method in Pattern Recognition, Proceedings of the First International Joint Conference on Pattern Recognition, Washington, D.C., October 30-November 1, 1973, pp. 25-32.
7. Karpiński, J., Michalski, R. S., A Digital System for Recognizing Handwritten Alphanumeric Characters (in Polish), Prace Instytutu Automatyki PAN, Warszawa, Zeszyt 35, 1966.
(An abbreviated English translation of the paper is available from the author.)
8. Michalski, R. S., On the Quasi-Minimal Solution of the General Covering Problem, Proceedings of the V International Symposium on Information Processing (FCIP 69), Vol. A3 (Switching Circuits), Yugoslavia, Bled, October 8-11, 1969.
9. Michalski, R. S., Synthesis of Minimal Structures of Combinational Automata and Recognition of Symmetry in Switching Functions, Proceedings of the Institute of Automatic Control, No. 92, Polish Academy of Sciences, Warsaw, 1971 pp. 155 (in Polish).
10. Michalski, R. S. and McCormick, B. H., Interval Generalization of Switching Theory; Proceedings of the Third Annual Houston Conference on Computer and System Science, Houston, Texas, April 26-27, 1971.

11. Michalski, R. S., A Geometrical Model for the Synthesis of Interval Covers, Report No. 461, Department of Computer Science, University of Illinois, Urbana, Illinois, June 24, 1971.
12. Lovby, Terje, Variable-Valued Logic Applied to Quality Control in a Clinical Laboratory, Master's Thesis, Department of Computer Science, University of Illinois, Urbana, 1974.



A Schematic Representation of
relations p, κ, τ

Figure 1