VARIABLE-VALUED LOGIC: SYSTEM VL1

by

Ryszard S. Michalski

Proceedings of the 1974 International Symposium on Multiple-Valued Logic, pp. 323-346, West Virginia University, West Virginia, May 29-31, 1974.

PROCEEDINGS OF THE

1974 INTERNATIONAL SYMPOSIUM ON MULTIPLE-VALUED LOGIC

Morgantown, West Virginia

May 29, 30, 31, 1974

Participating and Sponsoring Agencies:

COMPUTER SOCIETY OF THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS

MATHEMATICAL AND INFORMATION SCIENCE DIVISION OF THE OFFICE OF NAVAL RESEARCH

WEST VIRGINIA UNIVERSITY

Assigned IEEE Catalog Number 74CHO845 - 8C

Assigned ONR Identifying Number NR 048 - 616

Distribution and Copies of This Proceedings
Can Be Obtained Through The IEEE COMPUTER SOCIETY

UNCLASSIFIED

Price and Copies are Available from:

IEEECS
Post Office Box 639
Silver Spring, Md.
20901

IEEECS
c/o Reseda Van and Storage
7112 Canby Avenue
Reseda, California
91335

I.E.E.E.
305 East 46th Street
9th Floor
New York, N.Y.
10017

VARIABLE-VALUED LOGIC:

System VL_1

by

R. S. Michalski
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

ABSTRACT

The paper defines the concept of a variable-valued logic (VL) system and discusses in detail one specific VL system called VL_1.

The main motivation for the development of the VL system concept is to construct a well-defined, general formal system unifying various combinatorial problems and adequate, in particular, for problems of 'learning from examples' (problems which are of special interest to areas of pattern recognition and machine intelligence).

INDEX TERM:

Variable-valued logic, many-valued logic, switching functions, disjunctive normal forms, feature selection, classification rules, inductive systems.

specific sample facts or examples characterizing the objects. These descriptions can then be used, e.g., as simple rules for classifying or recognizing the objects. Another application can be to infer the simplest expression for a deterministic relationship existing among various objects, from examples of this relationship.

The set of operations which VL_1 uses to create such descriptions is limited to only a few basic logical operations and one arithmetic operation -- addition (which is used only in a restricted context, namely to express symmetric properties of functions with regard to their variables or inversed variables).

The original facts or examples, from which system can 'learn', have to be expressed in a special format: as sequences of properties taken from various finite (ordered or unordered) domains (i.e. a 'questionnaire format') or, in the most general case, in the form of VL_1 formulas.

The concept of a VL system and the definition of VL_1 were first introduced[2] at the IFIP Working Conference on Graphic Languages, Vancouver, Canada, May 1972.

In this paper we present an extended definition of VL_1 which includes certain new concepts, such as a symmetric selector, exception and separation operations and a combined VL_1 formula. We also give here a grammatical description of the syntax of VL_1 and describe various concepts relevant to the system: equivalence-preserving operations, merging and simplication rules, concept of minimality of VL_1 formulas under a lexicographic functional, a geometric model of VL_1 formulas.

A Reader interested in applications of the system and its computer implementation is referred to other papers[1,2,3].

Definition of the VL_1 system

VL_1 is a variable-valued logic system (X, Y, S, R_F, R_I)

where

X -- is a f.n. set of input variables whose domains are f.n. sets of elements called input semantic units. To be specific, we will assume, without loss of generality, that variables are

$$x_1 \quad x_2 \quad x_3 \quad \cdots \quad x_n \tag{2}$$

and their domains D_i are sets of non-negative integers with $d_1, d_2, \ldots, d_n$ elements, respectively:

$$D_i = \{0,1,2,\ldots,\not{d}_i\}\ , \quad i = 1,2,\ldots,n \tag{3}$$

where $\not{d}_i = d_i - 1$.

Y -- is a set consisting of one output variable whose domain is a f.n. set of elements called output semantic units. To be specific, we will assume, without loss of generality, that the variable is y and its domain D is a set of d non-negative integers:

$$D = \{0,1,2,\ldots,\not{d}\} \tag{4}$$

where $\not{d} = d-1$.

A possible interpretation of elements of D is interpret them as truth-values ('degrees of truth') or 'degrees of preciseness' of statements describing relations among values of variables x_i. Another, more general, interpretation of elements of D can be to interpret them as decisions which are made depending on values of x_i. Note that in this case elements of D may not have any 'natural' order, unlike in the previous interpretation.

S -- is the set of the following connecting symbols:

$$= \quad \neq \quad \geq \quad \leq \quad \underline{\vee} \quad \vee \quad \diagdown \quad \wedge \quad + \quad \neg \quad (\quad) \quad [\quad] \quad , \quad :$$

$\neg V$ is called the <u>inverse</u> of V (5)

$V_1 V_2$ is called the <u>product</u> (or <u>conjunction</u>) of V_1 and V_2 (6)

$V_1 \setminus V_2$ is called the <u>exception</u> V_2 <u>from</u> V_1 (or V_1 <u>except</u> <u>for</u> V_2) (7)

$V_1 \vee V_2$ is called the <u>sum</u> (or <u>disjunction</u>) of V_1 and V_2 (8)

$V_1 \mid V_2$ is called the <u>separation</u> of V_1 and V_2 (9)

R_I -- is a set of interpretation rules which assign to any VL_1 formula V a <u>value</u> $v(V) \in D$, depending on values of $x_1,\ldots,x_n$:

1. The value $v(c)$ of an element c, $c \in D$, is c, which is denoted :

$$v(c) = c \tag{10}$$

2. $$v([L \# R]) = \begin{cases} \not{a}, & \text{if } v(L) \# v(R) \\ 0, & \text{otherwise} \end{cases},$$

where

$v(L)$, $L \in \{\mathbf{x}, V_1\}$, is $v(V_1)$, i.e. value of wff V_1, if L is V_1, otherwise is $v(\mathbf{x})$, i.e. value of the sequence $\mathbf{x}$. Assuming that $\mathbf{x}$ is a sequence of literals $\tilde{x}_i$, $i \in I$, separated by '+', $v(x)$ is defined as:

$$v(x) = \sum_{i \in I} v(\tilde{x}_i), \tag{11a}$$

where

Σ denotes arithmetic sum

$$v(\tilde{x}_i) = \begin{cases} \dot{x}_i, & \text{if } \tilde{x}_i \text{ is } x_i \\ \not{a}_i - \dot{x}_i, & \text{if } \tilde{x}_i \text{ is } \neg x_i \end{cases} \tag{11b}$$

$\dot{x}_i$ denotes a value of variable x_i, $\dot{x}_i \in D_i$

$I \subseteq \{1, 2, \ldots, n\}$

<u>Example</u>: If values of variables x_1, x_2, x_3, x_4 are, respectively: 2, 0, 3, 4, and maximal elements in their domains: 4, 4, 5, 5, then:

$$v(x_2 + \bar{x}_3 + x_4) = 0 + (5-3) + 4 = 6$$

$v(R)$, $R \in \{\mathbf{c}, V_2\}$, is the sequence $\mathbf{c}$, if R is $\mathbf{c}$; otherwise $v(V_2)$,

DISCUSSION OF VL_1 FORMULAS

Examples of VL_1 and not VL_1 Formulas

A string of symbols from sets X, D and S, specified in the definition of VL_1, is a VL_1 formula if and only if it can be constructed by a finite number of applications of the formation rules R_F. Below are given some examples of VL_1 formulas, and, for comparison, some strings which are not VL_1 formulas.

VL_1 formulas:

$$3[x_1=0,3:5][x_3\neq 2,4] \vee 2[x_2\neq 3] \vee 1[x_4\geq 2] \tag{18}$$

$$([x_2=1:3] \vee [x_3\neq 4,6])(1 \vee [x_1=0,2,4] \vee [x_3=3]) \tag{19}$$

$$\neg(2[x_1=3] \vee 1)[x_3\neq 2] \mid 2([x_1+x_3=0,5]) \vee [x_2=2] \vee 1[x_2\neq 0] \tag{20}$$

Not VL_1 formulas:

$$([x_1=0,2:5] = 3)[x_1\neq 1] \vee 2[[[x_1=0] \vee 1 = [x=1]]]=2] \tag{21}$$

$$3[x_3\neq 0:3] \vee \neg 2[[x_3=0,1,3,5] = [x_3=2]] \vee x_3 \tag{22}$$

$$[2[x_1+x_4=0] \vee 1 = x_1] \vee 1[2,3=2[x_2\neq 2]] \mid 3[x_2+x_4\leq 3] \tag{23}$$

String (21) is not a VL_1 formula because the form $[x_1=0,2:5] = 3$ does not satisfy the definition of a selector (should be enclosed in [ ]). (22) is not a VL_1 formula because a variable standing alone is not a wff. (23) is not a VL_1 formula because x_1 is on the right-hand side of =, and also because the string '2,3' is on the left-hand side of = .

Event Space

The interpretation rules R_I assign to any VL_1 formula a value -- an element of set D, depending on the values of $x_1, x_2, \ldots, x_n$ -- elements of sets $D_1, D_2, \ldots, D_n$. Thus, the interpretation rules interpret VL_1 formulas as expressions of a function:

$$f: D_1 \times D_2 \times \ldots \times D_n \rightarrow D \tag{24}$$

where x denotes cartesian product and $\rightarrow$ 'mapping into'

Examples of VL_1 Formulas Interpretation

Recall the formula (18): $3[x_1=0,3:5][x_3\neq 2,4] \vee 2[x_2\neq 3] \vee 1[x_4\geq 2]$. This formula can be interpreted as an expression of a function:

$$f: E(6, 4, 5, 5) \rightarrow \{0, 1, 2, 3\} \tag{28}$$

assuming that $c(D_1)=6$, $c(D_2)=4$, $c(D_3)=5$, $c(D_4)=4$. The formula is assigned value 3 (briefly, <u>has value</u> 3), if selectors $[x_1=0,3:5]$ and $[x_3\neq 2,4]$ are satisfied, i.e., if x_i accepts value 0, 3, 4 or 5 and x_3 value 0, 1 or 3. (Thus, the value 3 of the formula does not depend on values of x_2 and x_4.) The formula has value 2 if the previous condition does not hold and the selector $[x_2\neq 3]$ is satisfied (recall that $v(V_1 \vee V_2) = \max\{v(V_1), v(V_2)\}$). The formula has value 1, if both of the previous conditions do not hold and the selector $[x_4\geq 2]$ is satisfied. If none of the above conditions hold, the formula has value 0.

Consider a formula:

$$2[x_2+x_3+x_4=2,3] \vee 1[x_1=0] \tag{29}$$

The formula has value 2 for all events in which values of variables x_2, x_3 and x_4 sum up to 2 or 3, except for such events with this property in which the value of variable x_1 is 0. For the latter events, and only for them, the formula has value 1. For all the other events the formula has value 0. Note that the function expressed by formula (29) is <u>symmetric</u> with regard to variables $\{x_2,x_3,x_4\}$, i.e. these variables can be exchanged one for another without changing value of the function (assuming that domains of these variables are of the same cordinality).

VL Functions

In describing objects (or processes) we usually deal with functions which may have an unspecified value for certain events, that is with functions :

expressed by a single VL_1 expression by extending the set X with an additional variable y whose domain is {1, 2, ..., m} and assuming that

$$D = \max({}^1D, {}^2D, \ldots, {}^mD).$$

The role of the variable y is expressed by an additional interpretation rule:

4. $v(V[y \# \mathbf{c}])$ is interpreted as follows:

the value $v(V)$ is given to functions $\{{}^yf \mid y \# \mathbf{c}\}$.

For example, if V_1, V_2 and V_3 are VL_1 formulas which do not include the variable y, then

$$V_1[y = 1,2,3] \vee V_2[y = 3,4] \vee V_3[y = 1,3] \tag{35}$$

in interpreted as an expression of a function $\mathbf{f} = ({}^1f, {}^2f, {}^3f, {}^4f)$, where expressions for functions ${}^kf_1$ $k = 1, 2, 3, 4$ are:

$${}^1f:\ V_1 \vee V_3,\quad {}^2f:\ V_1,\quad {}^3f:\ V_1 \vee V_2 \vee V_3,\quad {}^4f:\ V_2 \tag{36}$$

The selector $[y = \mathbf{c}]$ is called a <u>function selector.</u> A VL_1 formula which includes function selectors is called a <u>multiple-output VL_1 formula.</u> We extend a function $\mathbf{f}$ (33) to an incompletely specified function by assuming that:

$$f:\ \mathbf{E} \longrightarrow ({}^1D \cup \{*\}) \times ({}^2D \cup \{*\}) \times \ldots \times ({}^mD \cup \{*\}) \tag{37}$$

<u>Special Classes of VL_1 Formulas</u>

We will define some special classes of VL_1 formulas (disjunctive and conjunctive simple VL_1 formulas, cyclic formulas and interval formulas) which are of special interest for applications, because of the simplicity of their interpretation, evaluation and synthesis. We will start with defining some auxiliary concepts.

Definition 8. A VL_1 formula which is a conjunction of simple clauses is called a conjunctive simple (or normal) VL_1 formula and denoted CVL_1. (For example, (19) is a CVL_1.)

Definition 9. A VL_1 formula which is a DVL_1 or CVL_1 is called a simple (or normal) VL_1 formula.

Theorem 1. For any VL function there exists at least one DVL_1 and at least one CVL_1 formula, which are expressions of this function.

Proof. See paper[7].

Specification of formation rules as rewriting rules of a contex-free grammar

Below are given rewriting rules of a contex-free grammar (together with a corresponding terminology), which are an alternative way of expressing formation rules R_F of the VL_1. They are equivalent to R_F with the exception for sequences **x** and **c**. In R_F, **x** and **c** were defined as sequences of different elements while here they can include identical elements (**x** and **c**, as defined in R_F, cannot be expressed by a contex-free rewriting rules). A bar '|' and '::=' are used as metalanguage symbols for describing rewriting rules.

Formula	$V ::= \mathcal{L} \mid V \mid \mathcal{L}$
Clause	$\mathcal{L} ::= \mathcal{P} \mid \mathcal{L} \vee \mathcal{P}$
Phrase	$\mathcal{P} ::= T \mid \mathcal{P} \veebar T$
Term	$T ::= \mathcal{C} \mid T \wedge \mathcal{C} \mid T\mathcal{C}$
Component	$\mathcal{C} ::= c \mid S \mid \neg V \mid (V)$
Selector	$S ::= [L \# R]$
Left part (referee)	$L ::= \mathbf{x} \mid V$
Right part (reference)	$R ::= \mathbf{c} \mid V$
Selector relation	$\# ::= = \mid \neq \mid \geq \mid \leq$

All operations on V_1 and/or V_2 which preserve relation (40) are called <u>equivalence-preserving</u> <u>operations</u>.

<u>Examples of Equivalence-Preserving Operations on VL_1 Formulas</u>

Let $V \in \mathbf{V}(X,D)$.

$$V\not{d} \equiv V \qquad V \vee 0 \equiv V \qquad \text{(identity elements)} \tag{41}$$

$$V0 \equiv 0 \qquad V \wedge \not{d} \equiv \not{d} \qquad \text{(zero elements)} \tag{42}$$

$$\neg(c) \equiv \overline{c} \qquad \text{where } v(\overline{c}) = \not{d} - c \tag{43}$$

If $\#$ is $=$ or $\neq$ then:

$\neg[x_i \# c] \equiv [x_i \overline{\#} c]$, where '$\overline{\#}$' is $\{ {'\neq' \text{ if } '\#' \text{ is } '=' \atop '=' \text{ if } '\#' \text{ is } '\neq'}$

For example, $\neg([x_2=3:5]) \equiv [x_2\neq 3:5]$.

$\neg[x_i \geq c] \equiv [x_i \leq c^-]$, where c^- is equal c-1.

$\neg[x_i \leq c] \equiv [x_i \geq c^+]$, where c^+ is equal c+1.

Let V_1, V_2 and V_3 be elements of $\mathbf{V}(X,D)$.

$$V_1(V_1 \vee V_2) \equiv V_1 \qquad V_1 \vee V_1V_2 \equiv V_1 \qquad \text{(Absorption Laws)} \tag{44}$$

$$V_1(V_2 \vee V_3) \equiv V_1V_2 \vee V_1V_3 \quad V_1 \vee V_2V_3 \equiv (V_1 \vee V_2)(V_1 \vee V_3) \quad \text{(Distributive Laws)} \tag{45}$$

Let $\{V_j\}_{j\in J}$ be a subset of $\mathbf{V}(X,D)$.

$$\neg(\bigvee_{j\in J} V_j) \equiv \bigwedge_{j\in J} \neg(V_j) \tag{46}$$

$$\neg(\bigwedge_{j\in J} V_i) \equiv \bigvee_{j\in J} \neg(V_j) \qquad \text{De Morgan's Laws*} \tag{47}$$

<u>Examples:</u> $\neg(2[x_1=1:3][x_3\neq 2]) \equiv \overline{2} \vee [x_1\neq 1:3] \vee [x_3=2]$

where $v(\overline{2}) = \not{d}-2$

$$\neg(3[x_2=2,4] \vee 1[x_2=0][x_3\neq 1:3]) \equiv (\overline{3} \vee [x_2\neq 2,4])(1 \vee [x_2\neq 0] \vee [x_3=1:3])$$

Let $\mathbf{c}_1 \cup \mathbf{c}_2$ ($\mathbf{c}_1$ <u>merged</u> <u>with</u> $\mathbf{c}_2$) and $\mathbf{c}_1 \cap \mathbf{c}_2$ ($\mathbf{c}_1$ <u>intersected</u> <u>with</u> $\mathbf{c}_2$) denote compressed references obtained by set-theoretical summing and intersecting, respectively, the sets of elements in extended forms

* $\bigwedge$ and $\bigvee$ mean a conjunction and a disjunction of formulas, respectively.

a sum of interval selectors. (To see this, consider, e.g., the selector $[x_i=0,2:4,6]$. It can be expressed as a product $[x_i=0:6][x_i\neq 1][x_2\neq 5]$ or as a sum $[x_i=0]\vee[x_i=2:4]\vee[x_i=6]$.)

With regard to operations $\setminus$ and $|$ it follows from (15) and (17) that:

$$V_1 \setminus V_2 \equiv V_1[V_1V_2 = 0] \vee V_2$$

$$V_1 \mid V_2 \equiv V_1[V_2 = 0] \vee V_2[V_1 = 0]$$

MINIMAL VL_1 FORMULAS

A Planar Geometrical Representation of VL Functions and VL_1 Formulas

The merging and simplification rules (49, 50, 51, 52) are examples of rules which allow us to modify ('simplify') a given VL_1 formula without changing its interpretation (i.e., preserving the equivalence). There can be, in general, many different VL_1 formulas expressing the same VL function. Thus, a problem arises of how, for a given VL function, to construct the simplest (in some specified sense) VL_1 formula.

To illustrate this problem graphically, we will use a Generalized Logical Diagram (GLD) described in paper[4]. The GLD is a planar geometrical model of an event space $\mathbf{E}(d_1, d_2, \ldots, d_n)$, and provides a convenient representation of VL functions and VL_1 formulas. For example, Figure 1, shows a GLD representation of a VL function:

$$f: \mathbf{E}(4, 2, 3, 3) \rightarrow \{0, 1, 2, 3, *\}$$

Cells of the diagram correspond to events in $\mathbf{E}(4, 2, 3, 3)$. The correspondence is self-explanatory, e.g., cell 25 corresponds to event $e^{25} = (1, 0, 2, 1)$. Numbers on the top and the right of the diagram are the numbers of the events to which the cells correspond from which the user may select the most appropriate for his type of application. For computational feasibility, it is desirable, however, that the available functionals be expressed in one standard form. We found it convenient for computations and also useful in practice to assume that functionals are in the form:

$$A = \langle \mathbf{a}\text{-list}, \tau\text{-list} \rangle \tag{55}$$

where

a-list, called attribute (or criteria) list, is a vector $\mathbf{a} = (a_1, a_2, \ldots, a_\ell)$, where the a_i denote single- or many-valued attributes used to characterize DVL_1 formulas,

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

Functionals in the form (55) are called lexicographic functionals.

A DVL_1 formula V is said to be a minimal DVL_1 expression for f under functional A iff:

$$A(V) \overset{\tau}{<} A(V_j) \tag{56}$$

where

$A(V) = (a_1(V), a_2(V), \ldots, a_\ell(V))$

$A(V_j) = (a_1(V_j), a_2(V_j), \ldots, a_\ell(V_j))$

$a_i(V)$, $a_i(V_j)$ denote the value of the attribute a_i for formula V and V_j, respectively. V_j, $j=1,2,3,\ldots$--all irredundant DVL_1 expressions for f (a DVL_1 expression is called irredundant, if removing any term or selector from it makes it no longer an expression for f).

$\overset{\tau}{<}$ denotes a relation, called the lexicographic order with tolerance τ, defined as:

recognition, concept formation, inductive systems, etc. Program AQVAL/1, implementing a synthesis of minimal DVL_1 formulas, has already been successfully applied to a number of pattern recognition problems, such as synthesis of certain medical diagnostic procedures[1], design of a set of spacial filters for discrimination of non-uniform textures[1], 'discovering' simple classification rules[2], determination of simple rules for carbonate rocks classification in geology, and some others.

ACKNOWLEDGMENT

The author gratefully acknowledges the help he received from Professor Dennis Cudia in developing rewriting rules specifying the syntax of VL_1.

REFERENCES

1. Michalski, R. S., "AQVAL/1--Computer Implementation of a Variable-Valued Logic System and the Application to Pattern Recognition," Proceedings of the First International Joint Conference on Pattern Recognition, pp. 3-17, Washington, D.C., October 30-November 1, 1973.

2. _______, "A Variable-Valued Logic System as Applied to Picture Description and Recognition," GRAPHIC LANGUAGES, Proceedings of the IFIP Working Conference on Graphic Languages, Vancouver, Canada, May 1972.

3. _______, "Discovering Classification Rules by the Variable-Valued Logic System VL_1," Proceedings of the Third International Joint Conference on Artificial Intelligence, Theorem Proving and Logic, pp. 162-172, Stanford, California, August 20-24, 1973.

4. _______, "A Geometrical Model for the Synthesis of Interval Covers," Report No. 461, Department of Computer Science, University of Illinois, Urbana, Illinois, June 24, 1971.

5. Givone, P. D. and Snelsire, R. W., The design of multiple-valued logic systems, Rep. Department of Electrical Engineering, State University of New York at Buffalo, 1968.

6. Nutter, R. S., "Function simplification techniques for Postian multi-valued logic systems", Dissertation West Virginia University, 1971.

7. Michalski, R. S., Variable-valued logic system VL_1: I--elements of theory, to appear as a report of the Department of Computer Science, University of Illinois, Urbana, Illinois.

8. _______ and McCormick, B. H., "Interval generalization of switching theory", Proceedings of the Third Annual Huston Conference on Computer and System Science, Houston, Texas, April 26-27, 1971.