SYNTHESIS OF OPTIMAL AND QUASI-OPTIMAL VARIABLE-VALUED LOGIC FORMULAS

by

Ryszard S. Michalski

Proceedings of the 1975 International Symposium on Multiple-Valued Logic, pp. 76-87, Bloomington, Indiana, May 13-16, 1975.

PROCEEDINGS OF THE 1975 INTERNATIONAL SYMPOSIUM ON MULTIPLE-VALUED LOGIC

Indiana University, Bloomington, Indiana

May 13-16, 1975

Sponsoring Organizations

MATHEMATICAL AND INFORMATION SCIENCE DIVISION OF THE UNITED STATES OFFICE OF NAVAL RESEARCH
FORD INTERNATIONAL VISITORS EXCHANGE
INDIANA UNIVERSITY

Participating, Cooperating Organizations

COMPUTER SOCIETY OF THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS
ASSOCIATION FOR COMPUTING MACHINERY
SOCIETY FOR EXACT PHILOSOPHY

Assigned IEEE Catalog Number 75CHO959-7C
Assigned ONR Identifying Number NR 048-627

Price and Proceedings are Available from:
IEEE Computer Society
855 Naples Plaza, Suite 301
Long Beach, California 90803

SYNTHESIS OF OPTIMAL AND QUASI-OPTIMAL VARIABLE-VALUED LOGIC FORMULAS

R. S. Michalski
University of Illinois at Urbana-Champaign
U. S. A.

1. Introduction. It has recently been observed [1], [2], [3] that an extension of multi-valued logic in the form of a variable-valued logic system (VLS) suggests a new and promising model for decision theory, artificial intelligence, pattern recognition and related areas.

A VLS extends known many-valued logic systems (MLS) in two directions: (1) It assumes that propositions and variables in them take values not from the same domain, but from separate domains, whose size and structure is decided based on semantic- and problem-oriented considerations. (2) It generalizes some of the traditionally used operators (e.g., the concept of a 'selector' in the system [3] VL_1 is a generalization of 'literals' used in various MLS's), or adds new operators (e.g., symmetric selector or exception operator in VL_1).

In this paper we describe some ideas and algorithms for the synthesis of disjunctive simple formulas of the variable-valued logic system VL_1, which are optimal under a lexicographic functional [2].

The system VL_1 is the first and the simplest VL system whose practical applications have been investigated [2], [4]. Its definition and some formal properties have been described in paper [3], therefore we will only briefly summarize here the notation and basic concepts pertinent to VL_1.

2. Summary of Notation and Basic Concepts.

$x_1 x_2 x_3 \dots x_n$	input variables with domains $D_1, D_2, \dots, D_n$, respectively. It is assumed* here that $D_i = \{0, 1, \dots, \not{d}_i\}$;
y	output variable with domain D. It is assumed** here that $D = \{0, 1, 2, \dots, \not{d}\}$;
$E(d_1, d_2, \dots d_n)$ or E	an event space defined as $D_1 \times D_2 \times \dots \times D_n$, where $d_i = \not{d}_i + 1$;
$[L \# R]$	a selector which takes the value $\not{d}$, if it is satisfied, otherwise the value 0. In the selector: L is a single variable x_i, or an arithmetic sum of variables or their inverses, or a VL_1 formula; # denotes one of the following relations: '=' '≠' '≤' '≥' ; R is a subset of the union of domains of variables occurring in L ;

* Sets D_i can be, in principle, any (ordered or unordered) sets [3].

** Set D can be, in principle, any linearly ordered set which has minimum and maximum elements.

VL_1 formula	consists of selectors and elements from domain D, linked by operators $\neg, \vee, \wedge, \setminus$;
$V, V_1, V_2, V_3, \ldots$	VL_1 formulas ;
$\neg V$	inverse of V defined as $\not{d} - V$;
$V_1 V_2$ or $V_1 \wedge V_2$	conjunction or minimum of V_1 and V_2 ;
$V_1 \setminus V_2$	V_1 except for V_2, defined as $V_1[V_2 = 0]$;
$V_1 \vee V_2$	disjunction or maximum of V_1 and V_2 ;
DVL_1 formula	(disjunctive simple formula) is a disjunction of terms, where a term is a conjunction of selectors and a constant from D .

3. Optimalization Criterion for VL_1 Formulas. A VL_1 formula is interpreted as an expression of a function (VL function):

$$f: \quad E(d_1, d_2, \ldots, d_n) \to D \quad . \tag{1}$$

If two different formulas express the same function then they are called semantically equivalent. A DVL_1 formula V is called an optimal formula under functional $A = \langle a\text{-list}, \tau\text{-list} \rangle$, among all the semantically equivalent DVL_1 formulas V_j, if

$$A(V) \overset{\tau}{<\cdot} A(V_j) \quad ,$$

where a-list, called attribute (or criteria) list, is a vector $a = (a_1, a_2, \ldots, a_\ell)$, where the a_i denote single- or many-valued attributes used to characterize DVL_1 formulas (e.g., number of terms, of selectors, total number of variables involved, etc.) ;

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

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

$a_i(V)$, $a_i(V_j)$ denote the value of the attribute a_i for formula V and V_j, respectively ;

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

$$A(V) \overset{\tau}{<\cdot} A(V_j) \text{ if } \begin{cases} a_1(V_j) - a_1(V) > T_1 \\ \text{or } |a_1(V_j) - a_1(V)| \le T_1 \text{ and } a_2(V_j) - a_2(V) > T_2 \\ \text{or } \ldots\ldots\ldots \\ \vdots \\ \text{or } \ldots\ldots\ldots\ldots\ldots \text{ and } a_\ell(V_j) - a_\ell(V) \ge T_\ell \end{cases} ;$$

$T_i = \tau_i (a_{i\max} - a_{i\min})$, $i = 1, 2, \ldots, \ell$;

$a_{i\max} = \max_j \{a_i(V_j)\}$, $a_{i\min} = \min_j \{a_i(V_j)\}$.

Note that if $\tau = (0, 0, \ldots, 0)$ then $\overset{\tau}{\leq}$ denotes the lexicographic order in the usual sense. In this case, A is specified just as $A = \langle$a-list$\rangle$. The optimality functional A is called a lexicographic functional.

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

4. Elements of Covering Theory. Algorithms for the synthesis of optimal VL_1 formulas to be described here are based on the results of covering theory [5]-[8] This theory deals with problems of expressing any arbitrary sets in a universe Ω by means of certain standard subsets called complexes.

A set $\mathcal{C} \subseteq 2^{\Omega}$ is called universe of complexes, if it satisfies:

(i) coverability criterion:

$$\forall e \in \Omega, \quad \exists C \in \mathcal{C}, \quad e \in C \tag{2}$$

(ii) separability criterion:

$$\forall e_1, e_2 \in \Omega, \quad \exists C_1, C_2 \in \mathcal{C}, \quad (e_1 \in C_1, \ e_2 \in C_2 \text{ and } C_1 \cap C_2 = \emptyset).$$

Let E_1 and E_2 be two subsets of Ω.

Definition 1. A cover $CV(E_1|E_2)$ of set E_1 against set E_2 is defined as a set of complexes, $\{C_i\}_{i \in I}$, such that

$$E_1 \cap \overline{E}_2 \subseteq \bigcup_{i \in I} C_i \subseteq E_1 \cup \overline{E}_2 \tag{3}$$

where $\overline{E}_2 = \Omega \backslash E_2$.

We will assume here that Ω is an event space $\mathbf{E} = D_1 \times D_2 \times \ldots \times D_n$ and consider two universes of compleses: 1) $\mathbf{R}$, universe of cartesian complexes, R, defined as:

$$R_j = \bigcap_{i \in I} \{x_i = \alpha_i\} \tag{4}$$

where $\{x_i = \alpha_i\}$, called a cartesian literal, is a set of all events $e = (x_1, x_2, \ldots x_i, \ldots, x_n) \in \mathbf{E}$, such that the value of x_i is an element of α_i, $\alpha_i \subseteq D_i$; 2) $\mathbf{L}$, universe of interval complexes, L_j, defined as:

$$L_j = \bigcap_{i \in I} \{x_i = a_i : b_i\} \tag{5}$$

where $\{x_i = a_i : b_i\}$, called an interval literal, is a set of all events e such that the value of x_i is between a_i and b_i, inclusively.

Following are definitions of a few concepts necessary for understanding the principle of disjoint stars and the cover synthesis algorithms discussed later.

Let E, E_1, E_2 be event sets, i.e., subsets of E.

Definition 2. The cartesian (interval) root, $\sqrt{E}$ ($\dot{\sqrt{E}}$), of E is the set of all maximal cartesian (interval) complexes included in E.

$$\sqrt{E} = \{R \in \mathbf{R} \mid R \subseteq E \text{ and } \nexists R' \subseteq E, \ R \subseteq R'\} \tag{6}$$

The concepts of a star and an extension, to be defined below, will be modified by the adjective 'interval', whenever the root of an event set occurring in the definition of these concepts is an interval root.

Definition 3. The cartesian star or, simply, star $G(E_1|E_2)$ of E_1 against E_2 is defined:

$$G(E_1|E_2) = \{R \mid R \in \sqrt{\overline{E}_2} \text{ and } R \cap E_1 \neq \emptyset\} \tag{7}$$

The subset of $G(E_1|E_2)$ consisting of complexes C which cover entirely E_1, i.e., $E_1 \subseteq R$, is called the covering star $CG(E_1|E_2)$ of E_1 against E_2.

Lemma 1. $$G(E_1|E_2) = \sqrt{\overline{E}_2} \setminus \sqrt{\overline{E}_1 \cap \overline{E}_2} \tag{8}$$

Proof. The set $\sqrt{\overline{E}_2}$ can be partitioned into two classes of complexes: (1) a class of complexes which intersect E_1, and (2) a class of complexes which do not intersect E_1.

Class (1) is clearly $G(E_1|E_2)$. It is easy to see that class (2) can be expressed as

$$\sqrt{E \setminus (E_1 \cup E_2)} = \sqrt{\overline{E}_1 \cup \overline{E}_2} = \sqrt{\overline{E}_1 \cap \overline{E}_2} \tag{9}$$

which ends the proof. ■

Definition 4. The cartesion extension or, simply, the extension $E_1 \rightarrowtail E_2$, of the event set E_1 against event set E_2 is defined as:

$$E_1 \rightarrowtail E_2 = \bigcup \{R \mid R \in G(E_1|E_2)\} \tag{10}$$

The extension $E_1 \rightarrowtail \overline{E}_2$ is called the extension of E_1 in E_2 and denoted $^{*}E_1 \vdash E_2$. If F is a family of sets then the set-theoretic union of the sets of this family is denoted by $F^{\cup}$. Thus, $E_1 \rightarrowtail E_2 = (G(E_1|E_2))^{\cup}$, or, simply, $G^{\cup}(E_1|E_2)$.

If $E_1 \subseteq E_2$, then obviously $G(E_1|E_2) = \emptyset$, and if $E_1 \cap E_2 \neq \emptyset$ then $CG(E_1|E_2) \neq \emptyset$. Let e_1, e_2 be events outside of an event set E, and $G(e_1|E)$ and $G(e_2|E)$ denote the stars of e_1 and e_2 against E, respectively.

The stars $G(e_1|E)$ and $G(e_2|E)$ are disjoint if they share no common complexes (though the complexes in the stars may intersect). Let G^r be a family of pairwise disjoint stars, $G(e|E_2)$, $e \in E^r$, $E^r \subseteq E_1$, $E_1 \cap E_2 = \emptyset$. Let MCV be a cover $CV(E_1|E_2)$ which has the minimum number of complexes, i.e., an optimal cover under functional A = <# of complexes>. Let c(G) and c(M) denote cardinalities of G and M, respectively.

Theorem 1. (Principle of disjoint stars)

$$c(MCV) \geq c(G) \tag{11}$$

Proof. Stars $G(e|E_2)$, $e \in E^r$, are disjoint, therefore there does not exist a complex which can cover more than one event from E^r. Consequently, any cover $CV(E^r|E_2)$ will have to include at least c(G) complexes. Since $E^r \subseteq E_1$, then any cover $CV(E_1|E_2)$, thus also MCV, has to have at least c(G) complexes. ■

The theorem is true for any family of disjoint stars, thus also for a family of maximum cardinality, which, obviously, gives the most desired lower bound. The principle of disjoint stars has been first [5] formulated by Michalski in 1969 and used for developing a new approach for optimal cover synthesis [9], [10].

*In papers [6], [7] $\vdash$ is denoted by $\smile$.

The most important algorithm, from the viewpoint of applications, produced by this approach is the 'quasi-minimal [5] algorithm' A^q. This algorithm has been used as a basis for optimization of VL_1 expressions. It has the following important features:

1. It produces a quasi-minimal cover (minimal or approximately minimal) of a set against another set in a computationally very efficient way.

2. It produces an estimate Δ of the maximal possible distance (in the number of complexes) between the obtained cover, M^q, and minimal one, M:

$$c(M^q) - c(M) \stackrel{\leq}{\sim} \Delta \qquad (12)$$

The estimate Δ is computed as the difference between the number of complexes in the obtained cover and the number of disjoint stars which were generated during the execution of the algorithm. The estimate, as well as the cover itself, can be improved by repeating the algorithm.

3. The algorithm is 'robust', by which is meant that its computational complexity can be easily controlled and kept approximately constant, independent of the combinatorial complexity of the problem. That is, if the problem is simple, then the algorithm will function in a most optimal way, producing a solution which is optimal or very close to optimal under given functional A. But if a problem is combinatorially very complex, the algorithm will function in a less optimal way, but still will give a solution in a reasonable time (though the solution may be farther from the optimum).

There are many algorithms which share the first feature with the algorithm A^q. One of the most recent and advanced is described in paper [10]. There are, however, to the author's knowledge, no algorithms which also display the other two feature

5. Algorithm A^q. The basic idea of the algorithm A^q for the synthesis of a cover $CV(\overline{E_1|E_2})$ or set E_1 against set E_2, is to generate consecutive disjoint stars $G(e|E_2)$, $e \in E_1$, and to select from each star the best complex L^q ('quasi-extremal') according to an optimality criterion. This criterion can be specified as a functional $A = \langle a\text{-list}, T\text{-list}\rangle$, in which the a_i are attributes of complexes whose minimum is most desired from the viewpoint of the optimality criterion for the whole cover. For example, if the first requirement for a solution is that the cover should have the minimum number of complexes, then the first attribute on the attribute list, a-list, could be an inverse of the number of events in E_1 covered by a given complex. Part I of the algorithm terminates, when no more disjoint stars can be generated. If, at this moment, the set of events remaining to be covered (current value of E_1) is not empty, then Part II is executed. New stars (not disjoint this time) are generated and quasi-extremals determined, in a similar way as in Part I, until all events of the set E_1 are covered. To keep the algorithm within reasonable computational time limits even for very complex problems, two parameters are used: maxstar (MS) and cutstar (CS). Their role can be described as follows: If in the process of a star generation, a number of complexes at any moment is larger than the specified limit MS, then the set of complexes is cut down to only CS complexes, which are most desirable from the viewpoint of the assumed optimality functional A. If any of the stars is 'cut' in the execution of Part I, and it cannot be proved that the union of generated complexes for this star is equal $G^U(e|E_2)$, then the computed Δ ceases to be a true estimate. (1

The flowchart of the algorithm A^q is given in Fig. 1. The selection of events e_1 from E_p in Part I, and from E_1 in Part II, can be done arbitrarily (e.g., randomly) or according to some algorithm which may depend on the information about the function obtained from the previous application of the algorithm for that function (see, e.g., paper [11] which describes an adaptive interactive synthesis of logical formulas).

6. Star Generation. The most important and difficult part of A^q is generation of stars $G(e_1|E_2)$. We will present here one of a few algorithms developed for this purpose.

This algorithm is based on the following theorem:

Theorem 2. The union of complexes in a star $G(e|E)$ is equal to an intersection of extensions of event e in complements of events of E:

$$G^{\cup}(e|E) = \bigcap_{k=1}^{g} (\{e\} \vdash \{\overline{e_k}\}). \tag{13}$$

where $E = \{e_1, e_2, \dots, e_g\}$.

Proof. The proof is given in paper [6].

In order to obtain the star $G(e|E)$, it is necessary to express the right part of (13) as a union of maximal complexes. This can be done in the following steps:

1. Determine $S_k = \{e\} \vdash \{\bar{e}_k\}$ for each $e_k \in E$.

Suppose $e_k = \{x_1 = a_1\}\{x_2 = a_2\} \dots \{x_n = a_n\}$ (14)

then $\bar{e}_k = \{x_1 \neq a_1\} \cup \{x_2 \neq a_2\} \cup \dots \cup \{x_n \neq a_n\}$. By applying the following (15)

properties [6]:

$$E \vdash (R_1 \cup R_2 \cup \dots) = (E - R_1)(E_2 - R_2) \dots \tag{16}$$

$$E \vdash R = \begin{cases} R, & \text{if } R \cap E \neq \emptyset \\ \emptyset, & \text{otherwise} \end{cases} \tag{17}$$

represent each S_k as a union of complexes: $S_k = \{R_{k1} \cup R_{k2} \cup \dots\}$. (18)

2. From step 1 we have

$$G^{\cup}(e|E) = \bigcap_{k=1}^{g} S_k \quad . \tag{19}$$

By multiplying S_k by each other and applying absorption laws, the right side of (19) is transformed into an irredundant union of maximal complexes:

$$G^{\cup}(e|E) = R_1 \cup R_2 \cup R_3 \cup \dots \quad . \tag{20}$$

Thus

$$G(e|E) = \{R_i\}_{i=1,2,\dots} \quad . \tag{21}$$

7. Synthesis of Optimal VL_1 Formulas from Event Sets. A VL function f can be specified by a family of event sets:

$$F^d, F^{d-1}, \dots, F^1, F^0) \tag{22}$$

such that

$$F^k = \{e|f(e) = k\} \quad .$$

If $\bigcup_{k=0}^{d} F^k \neq E$ then f is incompletely specified. By an expression of an incompletely specified function is meant any expression which can assign a value from D to events $e \in E \setminus \bigcup_{k} F^k$ (called *-events or DON'T CARE).

We will show that an optimal VL_1 expression for f under a functional A can be constructed by determining a family of optimal covers under A of certain event sets against other sets.

Let $OC(E_1 | E_2)$ denote an optimal cover of E_1 against E_2 under functional A. An algorithm for constructing an optimal VL_1 expression for f consists of the following steps:

1. Determine optimal covers under A:

$$OC^{d} = OC(F^{d} / F^{d-1} \cup F^{d-2} \cup \ldots \cup F^{0})$$
$$OC^{d-1} = OC(F^{d-1} / F^{d-2} \cup F^{d-3} \cup \ldots \cup F^{0})$$
$$\vdots$$
$$OC^{1} = OC(F^{1} / F^{2}) \quad .$$

2. Express each complex in covers OC^k, $k = d, d-1, \ldots, 1$, as a term, i.e., a conjunction of selectors. Conjunct terms corresponding to complexes in C^d with d, in C^{d-1} with $d-1, \ldots,$ in C^1 with 1.

3. The disjunction of thus obtained terms is an optimal VL_1 expression of f under A.

The validity of this algorithm can be clearly seen by observing that disjunction of terms means the maximum of their values, and therefore events of F^{k_1} can be treated as *-events for all covers C^k, $k < k_1$.

The synthesis of optimal cartesian covers is a 'polynomial complete' problem [12] and its precise solution may require, in a general case, an unfeasibly large enumeration of various possibilities. (A proof of this is similar to the proof by Zhuravlev [13] on the necessity of enumeration in the minimization of Boolean expressions.)

Consequently, the only realistic approach is to apply an algorithm which seeks an approximate solution, such as algorithm A^q, whenever the problem becomes untractable for an exact solution.

The above described algorithm for the synthesis of VL_1 expressions which uses algorithm A^q for cover synthesis has been implemented as a PL/1 program called AQVAL/1. A functional description of this program and examples of its application to selected pattern recognition proglems has been described in paper [2].

8. Synthesis of a Family of VL_1 Expressions. Suppose we are given a family of VL_1 functions

$$\{ {}_{j}f : E \rightarrow {}^{j}D \}_{j=1, 2, \ldots, m} \qquad (23)$$

whose input domain is E and output domains are ${}^{j}D$, $j = 1, 2, , \ldots, m$. This family can be treated as a function:

$$\mathbf{f} : \mathbf{E} \rightarrow {}^{1}D \times {}^{2}D \times \ldots \times {}^{m}D \quad . \qquad (24)$$

Each function ${}^{j}f$ can be specified by a family of sets:

$$\{F^{jk}\}_{k \in {}^{j}D} \tag{25}$$

such that $F^{jk} = \{e \mid f(e) = k\}$, $k \in {}^{j}D$.

Let us consider first a special case when:

(i) ${}^{j}D = \{0, 1\}$ for all j

(ii) F^{j1}, $j = 1, 2, \ldots, m$ are all pairwise disjoint.

This case describes many problems in the area of decision theory and pattern recognition. Specifically, F^{j1}, $j = 1, 2, \ldots, m$, can be interpreted as sets of events to which a decision class j is assigned. Each event in F^{j1} lists specific properties of one object of this class. Events in F^{j0}, $j \in \{1, 2, \ldots, m\}$, represent examples of objects not belonging to the given class ('negative examples'). Variables $x_1, x_2, \ldots, x_n$ are descriptors which are used to describe objects, their domains D_i are sets of values which the descriptors can accept in describing various objects of the universe of discourse.

Now, the problem may be to determine the simplest description, in some sense, of each decision class. If we assume that the class descriptions are to be expressed in terms of the VL_1 system, then the problem is to determine an optimal VL_1 expression of each ${}^{j}f$, according to an optimality functional reflecting practical needs.

By a VL_1 expression of ${}^{j}f$ is meant a VL_1 formula $V({}^{j}f)$, such that

$$V({}^{j}(f)) = \begin{cases} 1, & \text{if } e \in F^{j1} \\ 0, & \text{otherwise} \end{cases} . \tag{26}$$

In problems of this kind, sets F^{jk} usually constitute a very tiny part of the whole space E. Therefore, the process of construction of formulas involves a generalization of inputed information and, as such, is an inductive process.

All the given information consists of sets F^{jk}, therefore any assignment of specified decisions to *-events can happen to be wrong in the view of any new information. We accept here a 'simplicity criterion' which means that we seek an expression ('explanation') of the given incompletely specified function which is the 'simplest' among all possible expressions of this function.

We will consider 2 specific situations:

DC. ('Disjoint covers') When it is desirable that descriptions of individual classes are disjoint, which means that for any $e \in E$, only one class description is satisfied (i.e., only one $V({}^{j}f)$, $j = 1, 2, \ldots, m$, is equal to 1).

IC. (Intersecting covers') When it is required that class descriptions are disjoint only for events $e \in F^{j1}$.

Synthesis algorithms for both situations are given below.

DC: 1. Determine $DC_1 = OC(F^{11} / \bigcup_{j=2}^{m} F^{j1} \cup F^{10})$

2. Determine $DC_2 = OC(F^{21} / \bigcup_{j=3}^{m} F^{j1} \cup DC_1^{\cup})$

3. Determine $DC_3 = OC(F^{31} / \bigcup_{j=4}^{m} F^{j1} \cup F^{30} \cup \bigcup_{i=1}^{2} DC_i^{\cup})$

m. Determine $DC_m = OC(F^{m1} / F^{m0} \cup \bigcup_{i=1}^{m-1} DC_i^{\cup})$

m+1. The final step is to represent each cover C_j as a VL_1 formula.

As we see, the obtained covers DC_j depend on the order of sets F^{j1}, $j = 1, 2, \ldots, m$. To obtain order independent covers, one can first execute algorithm IC described below, and then 'subtract' from each cover all the other covers (where 'subtract' means the set-theoretical subtraction applied to individual complexes of the covers).

IC: 1. Determine $IC_1 = OC(F^{11} / \bigcup_{j=2}^{m} F^{j1} \cup F^{10})$

2. Determine $IC_2 = OC(F^{21} / \bigcup_{\substack{j=1 \\ j \neq 2}}^{m} F^{j1} \cup F^{20})$

$\vdots$

m. Determine $IC_m = OC(F^{m1} / \bigcup_{j=1}^{m-1} F^{j1} \cup F^{m0})$.

Now, let us consider a general case when the ${}^{j}D$, $j = 1, 2, \ldots, m$, can be any finite sets and there are no restrictions on F^{jk}.

If D_i, $i = 1, 2, \ldots, n$ and ${}^{j}D$, $j = 1, 2, \ldots, m$ are all $\{0, 1\}$, then f becomes a multiple output binary switching function. When ${}^{j}D$ are not binary, an important interpretation of f can be as a 'multivalued non-unique' decision function. Elements of ${}^{j}D$ can be interpreted as 'degree of truth' or 'confidence degree' that an event $e \in F^{jk}$, $k \in {}^{j}D$, should be assigned decision j. The function is not 'unique', because it can assign to an event more than one decision.

An optimality criterion for VL_1 expressions of f can be a functional $A = \langle a\text{-list}, \tau\text{=list} \rangle$, where elements of a-list are attributes which characterize whole set $\{V({}^{j}f)\}$ of VL_1 expressions of ${}^{j}f$.

A simple way of optimizing **f** is to treat it as one VL function:

$$f^0 : \mathbf{E} \times D_f \rightarrow D \tag{27}$$

Where $D_f = \{1, 2, \ldots, m\}$ is a domain of an additional input variable w and

$$D = \bigcup_{j=1}^{m} {}^{j}D$$

where $\cup$ means an ordered union of ${}^{j}D$, $j = 1, 2, \ldots, m$, i.e., a union of sets ${}^{j}D$, which is a linearly ordered set. Values of w are indexes indicating individual functions ${}^{j}f$.

Example. If V_1, V_2, V_3 are VL_1 expressions not involving variable w, then the expression

$$V_1[w=1, 2] \vee V_2[w=1, 3] \vee V_3[w=3, 4] \tag{28}$$

describes a family of VL_1 expressions:

$$\{V(^jf)\}_{j=1,2,3,4}$$

where

$$V(^1f) = V_1 \vee V_2$$
$$V(^2f) = V_1$$
$$V(^3f) = V_2 \vee V_3$$
$$V(^4f) = V_3 \quad .$$

9. Conclusion. We have described here various concepts and algorithms oriented toward synthesis of optimal and quasi-optimal disjunctive normal VL_1 formulas. We have shown that such a synthesis consists of determining optimal covers of various event sets against other events sets. We discussed only problems of synthesizing VL_1 formulas from event sets defining a given VL function. The methodology described here can be, however, easily extended for synthesizing optimal formulas from (arbitrary) VL_1 formulas, not just from event sets. This topic and, as well, other topics related to synthesis of VL_1 formulas (e.g., synthesis of formulas with symmetric selectors) are beyond the scope of this paper and will be described elsewhere.

Acknowledgment

The author expresses his gratitude to Mr. James Larson for his valuable comments and corrections of the original version of this paper. This paper was supported in part by National Science Foundation DCR74-03514.

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, Vancouver, Canada, May 1972.

2. 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.

3. 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.

4. 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.

5. 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 (in English).

6. Michalski, R. S. and B. H. McCormick, "Interval Generalization of Switching Theory," Proceedings of the Third Annual Houston Conference on Computer and System Science, Houston, Texas, April 26-27, 1971.

7. 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.

8. Michalski, R. S. and B. H. McCormick, "Cartesian Covers: A Class of Expressions for Sets and Relations," to appear.

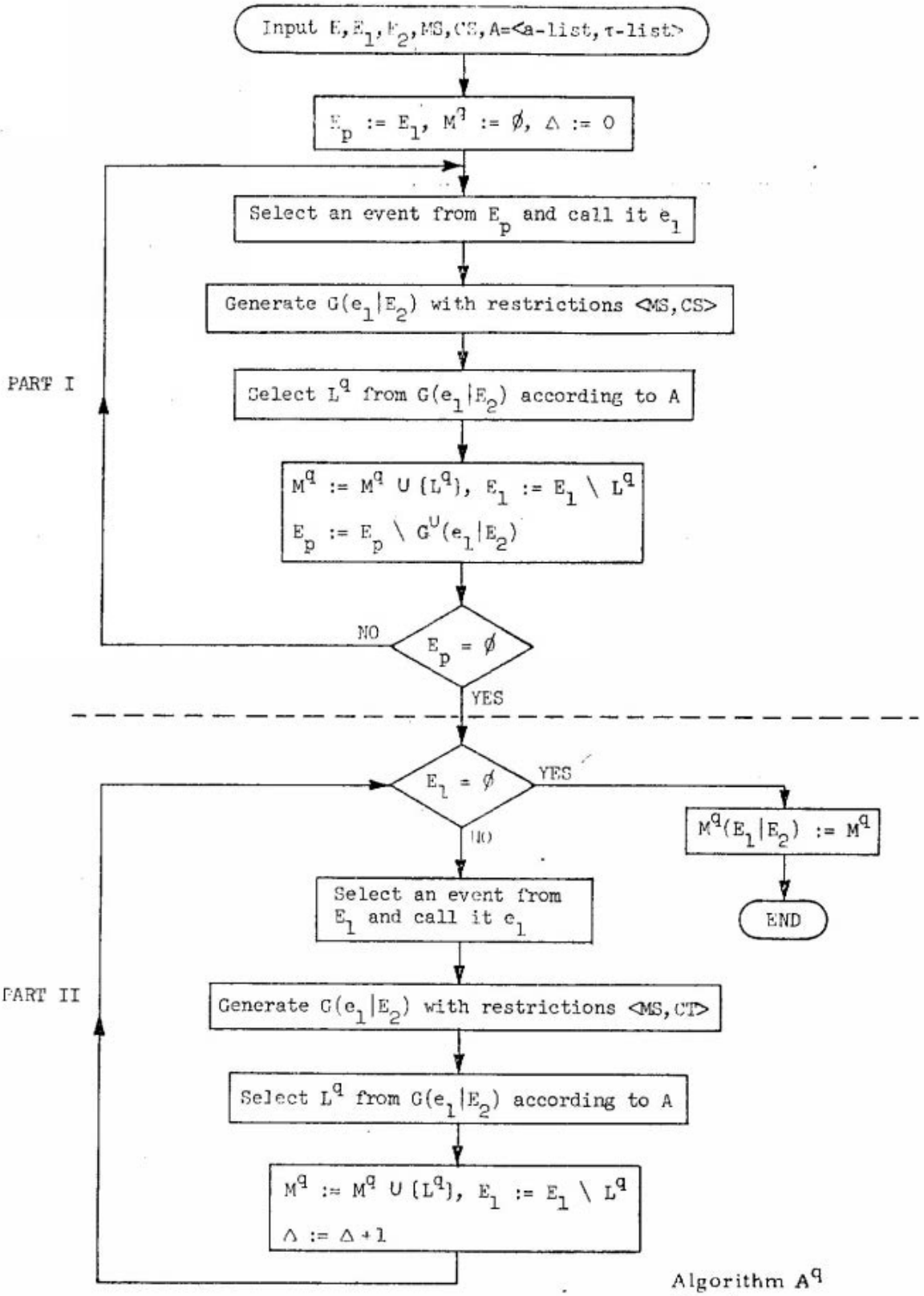
9. Michalski, R. S., "Synthesis of Minimal Forms and Recognition of Symmetry of Switching Functions," Proceedings of the Institute of Automatic Control, No. 92, Polish Academy of Sciences, Warsaw 1971 (in Polish).

10. Hong, S. J., R. G. Cain, and D. L. Ostapko, "MINI: A Heuristic Approach for Logic Minimization," IBM Journal of Research and Development, Vol. 18, No. 5, September 1974, pp. 443-458.

11. Michalski, R. S., "Automatic Synthesis of the Quasi-Minimal Multiple-Output Switching Circuits," VI International Symposium on Information Processing (FCIP 70), Yugoslavia, Bled, September 23-26, 1970.

12. Johnson, D. S., "Approximation Algorithms for Combinatorial Problems," Proceedings of the 5th Annual ACM Symposium on the Theory of Computing, 1973, pp. 38-49.

13. Zhuravlev Yu.I., "O nevozmozhnosti postroeniya minimalnykh normalnykh form funktsii algebry logiki v odnom klasse algoritmov," Doklady AN SSSR, Vol. 132, No. 3, 1960, pp. 504-506.



Algorithm A^q
Fig. 1