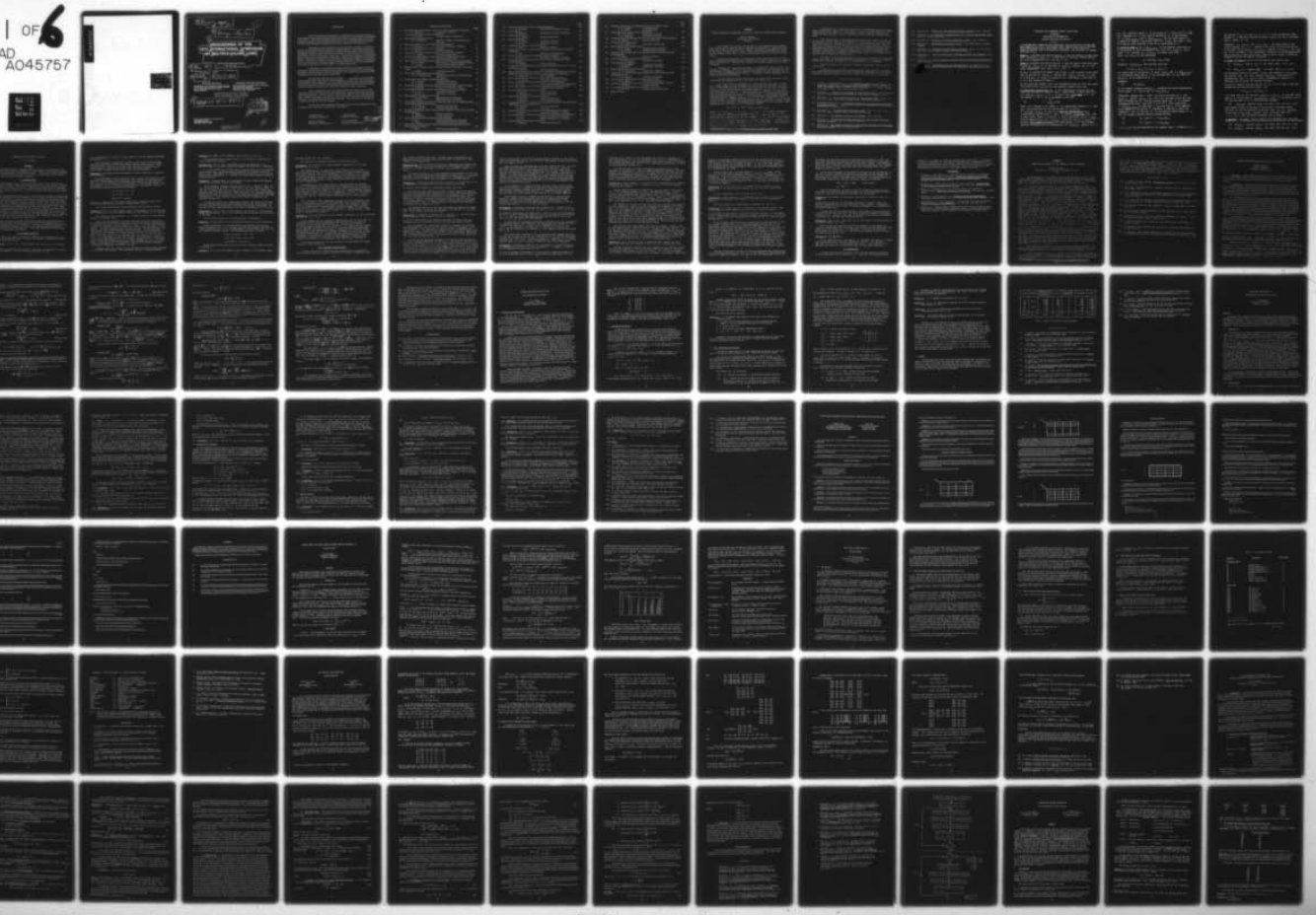
AD-A045 757 INDIANA UNIV BLOOMINGTON DEPT OF COMPUTER SCIENCE F/G 9/2
PROCEEDINGS OF THE 1975 INTERNATIONAL SYMPOSIUM ON MULTIPLE-VAL--ETC(U)
MAY 75 G EPSTEIN N00014-75-C-0449
UNCLASSIFIED MVL-75-001 NL



AD A045757

DDC
RECEIVED
NOV 1 1977
A

(9) Final rept.,

(10) George Epstein

(12)

(6)

PROCEEDINGS OF THE 1975 INTERNATIONAL SYMPOSIUM ON MULTIPLE-VALUED LOGIC

Indiana University, Bloomington, Indiana
May 13-16, 1975

(11) May 75

(12) 482 p.

(14) MVL-75-001

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

(15) N00014-75-C-0449

DDC
RECEIVED
NOV 1 1977
A

Price and Proceedings are Available from:
IEEE Computer Society
5855 Naples Plaza, Suite 301
Long Beach, California 90803

407210

FOREWORD

The 1975 International Symposium on Multiple-Valued Logic continues to bring together those having interests related to the theory and applications of multiple-valued logic. This fifth annual symposium extends the successful pattern set by the previous symposia at West Virginia University in 1974, at the University of Toronto in 1973, and at the State University of New York at Buffalo in 1972 and 1971.

Sponsorship of the Symposium was provided by the US Office of Naval Research, the Ford International Visitors Exchange Program, and Indiana University through the following: Office of Research and Development; Office of the Horizons of Knowledge Program; Mahlon Powell Lecture Series of the Department of Philosophy; Departments of Computer Science, Philosophy, Mathematics, Linguistics, Psychology, Physics, East Asian Languages and Literatures, History and Philosophy of Science, History, Sociology; the Center for Neural Science, East Asian Studies; the Linguistics Club, and the Indiana University Chapter of the Association for Computing Machinery. The help and financial aid of these units are greatly appreciated.

Cooperation and participation were provided by the Computer Society of the Institute of Electrical and Electronics Engineers, the Association for Computing Machinery, and the Society for Exact Philosophy. The support and services of these organizations are likewise greatly appreciated.

We thank all those who have contributed to the symposium, and are especially grateful to Dr. David Rine of West Virginia University, Mr. Joel Trimble of the US Office of Naval Research, Mr. True Seaborn of the Computer Society of the Institute of Electrical and Electronics Engineers, and Mr. Wayne Craig of the Indiana University Conference Bureau, for their encouragement, advice, and help.

This symposium would not be possible without the help of all those who submitted papers, the referees who contributed their voluntary time to ensure strict refereeing standards, and those authors whose works appear here in these Proceedings. These works appear here exactly as received, each revised by the authors according to our refereeing and publication standards. We are grateful to the authors for their extra efforts with this revising.

We heartily thank the Session Chairmen and all those in attendance at this symposium for their time, consideration, and support.

George Epstein
Symposium Chairman

J. Michael Dunn
Program Chairman

Stuart C. Shapiro
Program Vice-Chairman

Nino Cocchiarella
Program Vice-Chairman

Letter on copyright is on file

A

TABLE OF CONTENTS

			Page
1.	"Multiple-Valued Algorithmic Logics as a Tool to Investigate Programs"		1
	H. Rasiowa	University of Warsaw	
2.	"Lattices with Greatest (Least) Chain Base"		4
	T. Traczyk	Warsaw Technical University	
3.	"Reducibility of Post Functions"		8
	E. G. DuCasse	Brooklyn College, City University of New York	
4.	"Recent Developments in the Theory of Post Algebras"		18
	P. Dwinger	University of Illinois - Chicago	
5.	"Some Further Properties of the Pi-Logics"		20
	V. Pinkava	Severalls Hospital	
6.	"Ternary Two-Place Functions that are Complete with Constants"		27
	J. C. Muzio	University of Manitoba	
7.	"Functional Completeness in Heterogeneous Multiple-Valued Logics"		34
	I. G. Rosenberg	Université de Montréal	
8.	"The Linearity Property and Functional Completeness in M-Valued Logic"		44
	H. A. Ellozy	IBM Thomas J. Watson Research Center	
	Y. N. Patt	North Carolina State University	
9.	"Second Order and Higher Order Universal Decision Elements in m-Valued Logic"		53
	J. Loader	Brighton Polytechnic	
10.	"The Logical Foundations of Microlanguages"		58
	T. C. Wesselkamper	Virginia Polytechnic Institute and State University	
11.	"Multivalued Logic Design and Postian Matrices"		67
	R. S. Ledley	Georgetown University	
	H. K. Huang	Georgetown University	
12.	"Synthesis of Optimal and Quasi-Optimal Variable-Valued Logic Formulas"		76
	R. Michalski	University of Illinois - Urbana	
13.	"A Generalized Boolean Algebra and Its Application to Logic Design"		88
	S. C. Lee	University of Houston	
	Y. Keren-Zvi	University of Houston	
14.	"Representation of Discrete Functions"		99
	J. P. Deschamps	MBLE Research Laboratory	
	A. Thayse	MBLE Research Laboratory	
15.	"A Computer-Oriented Heuristic Minimization Algorithm for Multiple-Output Multi-Valued Switching Functions"		112
	P. T. Cheung	Packard Instrument Inc.	
	D. M. Purvis	Packard Instrument Inc.	
16.	"Synehtsis of Multiple-Valued Logic Networks Based on Tree-Type Universal Logic Modules"		121
	T. Higuchi	Tohoku University	
	M. Kameyama	Tohoku University	
17.	"Associative Memories as Multipath Logic Switches"		131
	Y. Pao	Case Western Reserve University	
	J. Altman	Case Western Reserve University	
18.	"Associative and Multi-Valued Logic for Possible Improvements in Some X-Ray Image Processing"		146
	D. Rine	West Virginia University	

			Page
19.	"Applications of Fuzzy Logic to Medical Diagnosis"		162
	H. Wechsler	University of California - Irvine	
20.	"Local Logics"		175
	R. Bellman	University of Southern California	
21.	"Fuzzy Modal Logic"		176
	P. K. Schotch	Dalhousie University	
22.	"Possible Automata"		183
	B. R. Gaines	University of Essex	
	L. Kohout	University of London	
23.	"Lukasiewicz Logic and Fuzzy Set Theory"		197
	R. Giles	Queen's University	
24.	"Conjectures on Many-Valued Logic, Regions, and Criteria for Conflict Resolution"		212
	S. Gale	University of Pennsylvania	
25.	"Free n-Valued Lukasiewicz Algebras without Involution"		226
	R. Cignoli	University of Illinois - Chicago	
26.	"On the Algebras Corresponding to the n-Valued Lukasiewicz-Tarski Logical Systems"		234
	R. Grigolia	Tbilisi University	
27.	"A Theorem on the Finiteness of the Degree of Maximality of the n-Valued Lukasiewicz Logic"		240
	R. Wójcicki	Wroclaw University	
28.	"Matrix Representation for the Dual Counterparts of Lukasiewicz n-Valued Sentential Calculi and the Problem of their Degrees of Maximality"		252
	G. Malinowski	Lódź University	
29.	"Some Applications of a General Theory of Digraph Measures"		262
	J. C. Hansen	University of Missouri - Rolla	
30.	"Binary and Multiple-Valued Models of Binary Gate Networks"		277
	M. Yoeli	Technion-Israel Institute of Technology	
31.	"A Ternary Algebra for Probability Computation of Digital Circuits"		280
	S. C. Hu	Cleveland State University	
32.	"Ternary Logic System Based on T-Gate"		290
	T. Higuchi	Tohoku University	
	M. Kameyama	Tohoku University	
33.	"Bilineal Separability of Ternary Functions"		305
	J. Nazarala	Universidad de Chile	
	C. Moraga	Universität Dortmund	
34.	"Implementation of a Complete Ternary Algebra with Elementary Operators - Application to Ternary Flip Flop"		316
	D. Etiemble	Paris VI Université	
	M. Israel	C. N. A. M.	
35.	"Some Multi-Valued Approaches to Two-Valued Switching Problems"		330
	G. Metze	University of Illinois - Urbana	
36.	"On the Efficiency of Ternary Algorithms for Multiplication and Division"		331
	A. Barak	Hebrew University of Jerusalem	
	E. Aron	Hebrew University of Jerusalem	
37.	"Hybrid Logic (A Fast Ternary Adder)"		344
	C. Moraga	Universität Dortmund	

			Page
38.	"A Design Technique for an Integrable Ternary Arithmetic Unit"		359
	H. T. Mouftah	Université Laval	
	I. B. Jordan	Université Laval	
39.	"Threshold Logic in Fast Ternary Multipliers"		373
	Z. G. Vranesic	University of Toronto	
	V. C. Hamacher	University of Toronto	
40.	"A Henkin-Type Completeness Proof for 3-Valued Logic with Quantifiers"		388
	H. Leblanc	Temple University	
41.	"A Useful Four-Valued Logic"		399
	N. D. Belnap, Jr.	University of Pittsburgh	
42.	"Compactness and ρ-Valued Logics"		400
	K. K. Hicken	Michigan State University	
	J. M. Plotkin	Michigan State University	
43.	"Finitely-Many-Valued Logics with Infinitely-Many-Valued Extensions: Two Examples"		406
	D. Ulrich	Purdue University	
44.	"Truth Functionality and Natural Deduction"		412
	J. D. McCawley	University of Chicago	
45.	"On Equivalential Algebras"		419
	J. K. Kabziński	Jagiellonian University	
	A. Wroński	Jagiellonian University	
46,	"Supervaluations in Two Dimensions"		429
	H. G. Herzberger	University of Toronto	
47.	"Similarity as a Theory of Graded Equality for a Class of Many-Valued Predicate Calculi"		436
	C. G. Morgan	University of Alberta	
48.	"On the Navya-Nyāya Logic of Property and Location"		450
	B. K. Matilal	University of Toronto	
49.	"A Survey of Studies on Applications of Many-Valued Logic in Japan"		462
	T. Kitahashi	Osaka University	
50.	"A Critical Survey of Many-Valued Logics 1966-1974"		468
	R. G. Wolf	Southern Illinois University	
51.	Editorial Notes.		475

SUMMARY

MULTIPLE-VALUED ALGORITHMIC LOGICS AS A TOOL TO INVESTIGATE PROGRAMS†

Helena Rasiowa
University of Warsaw
Poland

The attempt to develop the research in the mathematical programming theory led to various ideas, methods and approaches. Essential progress has recently been made in a logical approach to programming theory on the basis of algorithmic logic. These investigations have been initiated in papers [13], [14], [15] and developed in [2], [3], [4], [16] and others.

Formalized languages of algorithmic logic are extensions of first order predicate languages without quantifiers, however they contain another kind of quantifier. Programs are treated as certain expressions in these languages.

Formulas in formalized languages of algorithmic logic describe properties of programs. For instance the stop property, the correctness, the equivalence of programs can be expressed by means of such formulas and examined using metamathematical methods. Programs in these languages do not contain recursive procedures.

Various multiple-valued extensions of algorithmic logic have been formulated and examined in [6], [5], [8], [9], [10], [11].

The first kind of extensions is connected with the idea to treat "if...then...else" as a particular case of m-valued branchings, for all $m \geq 2$. For that purpose a mix-valued logic [12] is applied. Predicate calculi of this logic contain m-valued predicates for all $m \geq 2$, and each formula--with respect to the syntax--may be treated as a formula of m-valued logic for some m . This approach led to ω^+-valued algorithmic logic and simplified considerably the description of complicated programs, generalizing also "while...do...". On restricting the languages under consideration to n-valued formulas for $2 \leq n \leq m$, where m is an established integer, we obtain mix-valued algorithmic logics with logical values reduced to m possible. The semantics of programs and of the logics under consideration is related to the following algebra

$$P_\omega = (P_\omega, \vee, \cup, \cap, \Rightarrow, \neg, (d_i)_{0<i<\omega}, (e_i)_{0\leq i\leq\omega}) \ ,$$

where $P_\omega = (e_i)_{0\leq i\leq\omega}$, $e_0 \leq e_1 \leq \ldots \leq e_\omega = \vee$ is a chain isomorphic to the chain of ordinals not greater than ω , $(P_\omega, \vee, \cup, \cap, \Rightarrow, \neg)$ is a pseudo-Boolean algebra with the unit element $\vee = e_\omega$ and the zero element $\wedge = e_0$, i.e., $e_i \cup e_k = e_{max(i,k)}$, $e_i \cap e_k = e_{min(i,k)}$, $e_i \Rightarrow e_k = e_\omega$ for $i \leq k$, $e_i \Rightarrow e_k = e_k$ for $i > k$, $\neg e_i = e_i \Rightarrow e_0$, and moreover $d_i(e_k) = e_\omega$ for $i \leq k$, $d_i(e_k) = e_0$ for $i > k$.

† This paper will appear in the book Modern Uses of Multiple-Valued Logic.

A Hilbert style formalization and the completeness theorem for all these algorithmic logics have been given in [10]. Metamathematical investigations have been based on the theory of generalized Post algebras of order ω^+ which satisfy a finite representability condition for each element.

The crucial case of multiple-valued algorithmic logics is that of extended ω^+-valued, whose first version is proposed in [11] and which may be considered as an attempt to formalize the programming theory with recursive procedures. The application of ω^+-valued logic seems to play in this case an essential part. This approach to programming theory relates programs without recursive procedures to finite-control algorithms and recursive procedures to pushdown algorithms [1] (another treatment of recursive procedures on the ground of algorithmic logic [16] is based on a special kind of implicit definitions).

In order to express properties of procedures formulas of enumerable length are used. To metamathematical investigations the theory of generalized Post algebras of order ω^+, as examined in [7], is applied.

Various versions of extended ω^+-valued algorithmic logics can be considered.

Formulations of extended ω^+-valued algorithmic logics, including the syntax and the semantics of their formalized languages, will be given. Moreover, related problems as well as certain results will be discussed.

R E F E R E N C E S

[1] Blikle, A.; Mazurkiewicz, A. An algebraic approach to the theory of programs, algorithms, languages and resursiveness. Proc. Intern. Symp. and Summer school on Math. Found. of Comp. Sci., Warsaw, Jabłonna, August 21-27, 1972, CCPAS Reports, 1972.

[2] Kreczmar, A. Effectivity problems in algorithmic logic. Ph.D. Thesis Faculty of Mathematics and Mechanics, University of Warsaw, 1973.

[3] Mirkowska, G. On formalized systems of algorithmic logic. Bull. Ac. Pol. Sci., Ser. Sci. Math. Astron. Phys., 19 (1971), 421-428.

[4] Mirkowska-Salwicka, G. Algorithmic logic and its applications in the programming theory. Ph.D. Thesis, Faculty of Mathematics and Mechanics, University of Warsaw, 1972.

[5] Perkowska, E. On algorithmic m-valued logics, Bull. Ac. Pol. Sci., Ser. Sci. Math. Astron. Phys., 20 (1972), 717-719

[6] Rasiowa, H. On logical structure of programs, Ibid. 312-324.

[7] Rasiowa, H. On generalized Post algebras of order ω^+ and ω^+-valued predicate calculi, Ibid. 21 (1973), 209-219.

[8] Rasiowa, H. On logical structure of mix-valued programs and ω^+-valued algorithmic logic. ibid. 21 (1973), 451-458.

[9] Rasiowa, H. Formalized ω^+-valued algorithmic systems, Ibid. 559-565.

[10] Rasiowa, H. A simplified formalization of ω^+-valued algorithmic logic, Ibid. 22 (1974), 595-603.

[11] Rasiowa, H. Extended ω^+-valued algorithmic logic, Ibid. 605-610.

[12] Rasiowa, H. Mix-valued predicate calculi, to be published in Studia Logica.

[13] Salwicki, A. Formalized algorithmic languages. Bull. Ac. Pol. Sci., Ser. Sci. Math. Astron. Phys., 18 (1970), 227-232.

[14] Salwicki, A. On the equivalence of FS-expressions and programs, Ibid. 275-278.

[15] Salwicki, A. On the predicate calculi with the iteration quantifiers, Ibid. 279-285.

[16] Salwicki, A. Programmability and Recursiveness (an application of gorithmic logic to procedures), to appear in Dissertationes Mathe- ticae.

LATTICES WITH GREATEST (LEAST) CHAIN BASE

T.TRACZYK
INSTITUTE OF MATHEMATICS
Warsaw Technical University
Poland

1. Introduction. G.Epstein and A.Horn,[2], have recently discovered that Post algebras of order n are exactly those bounded distributive lattices which have unique n-term chain base. This, very nice and important characterization, may be obtainted by combining the following two lemmas (see also [5]).

Lemma 1. A bounded distributive lattice A has the greatest n-term chain base $0=e_0 \prec e_1 \prec \ldots \prec e_{n-1} = 1$ if and only if $ae_i \leqslant e_{i-1}$ implies $a \leqslant e_{i-1}$ for every complemented $a \in A$ and $i=1,\ldots,n-1$.

Lemma 2. Abounded distributive lattice A has the least n-term chain base $0=e_0 \leqslant e_1 \leqslant \ldots \leqslant e_{n-1} = 1$ if and only if $ae_i \leqslant e_{i-1}$ implies $ae_i = 0$ for every complemented $a \in A$ and $i = 1,\ldots,n-1$.

In fact, it follows from lemmas 1 and 2 that A has a unique n-term chain base if and only if $ae_i \leqslant e_{i-1}$ implies both $a \leqslant e_{i-1}$ and $ae_i = 0$, that is if and only if $ae_i \leqslant e_{i-1}$ implies $a = 0$, that means: if and only if A is a Post algebra of order n.

On the other hand lemmas 1 and 2 may be of some interest by their own, since they characterize P_o-lattices with greatest (least) chain base.

2. Preliminary definitions. Let A be a distributive lattice with the least and the greatest members 0 and 1, respectively. A chain sublattice $0 = e_0 \leqslant e_1 \leqslant \ldots \leqslant e_{n-1} = 1$ of A is said to be a chain base of A iff each member x of A can be expressed in the form

$$(1) \qquad x = \bigvee_{1 \leqslant i \leqslant n-1} x_i e_i$$

where x_i's belong to the center B of complemented elements of A and $x_i e_i$ is a shortening for the lattice meet of x_i and e_i.
If $x_1 \geqslant x_2 \geqslant \ldots \geqslant x_{n-1}$, we call (1) a monotone representation of x.
A distributive lattice with 0 and 1 is called a P_o-lattice, [3], iff it has a chain base. A P_o-lattice A with a chain base $e_0,e_1,\ldots,e_{n-1}$ will be denoted by $\langle A; e_0,\ldots,e_{n-1}\rangle$. A P_o-lattice $\langle A; e_0,\ldots,e_{n-1}\rangle$ is called a Post algebra of order n iff every element $x \in A$ has a unique monotone representation or, equivalently, iff $ae_i \leqslant e_{i-1}$ implies $a = 0$

for $i \geqslant 1$ and $a \in B$, where B is the center of A (see e.g. [1], [3]). The complement of a, $a \in B$, will be denoted by $\bar{a}$ in what follows. If $e = (e_i)_{0 \leqslant i \leqslant n-1}$ and $f = (f_i)_{0 \leqslant i \leqslant n-1}$ are two chain bases of a P_o-lattice A, then by definition $e \leqslant f$ iff $e_i \leqslant f_i$ for every i. This is a partial order relation in the set of chain bases of A.

3. Proof of lemma 1. Let $(e_i)_{0 \leqslant i \leqslant n-1}$ be the greatest chain base in a P_o-lattice A with the center B. Let $a \in B$ and $ae_i \leqslant e_{i-1}$ for a fixed $i \geqslant 1$. Setting $f_i = e_i \vee ae_{i+1}$ we obtain $\bar{a}f_i = \bar{a}e_i$. Since $ae_i = ae_{i-1}$ by our hypothesis,

$$e_i = ae_i \vee \bar{a}e_i = ae_{i-1} \vee \bar{a}f_i \tag{2}$$

Evidently $e_i \leqslant f_i \leqslant e_{i+1}$. This and (2) imply that

$$0 = e_o, \ldots, e_{i-1}, f_i, e_{i+1}, \ldots, e_{n-1} = 1$$

is a chain base of the lattice A. Since $f_i \geqslant e_i$ and $e = (e_i)_{0 \leqslant i \leqslant n-1}$ is the greatest chain base of A, we conclude that $f_i = e_i$, that is $ae_{i+1} \leqslant e_i$ by the definition of f_i.

Therefore

$$ae_{i+1} \leqslant e_{i-1} \tag{3}$$

by the assumed inequality $ae_i \leqslant e_{i-1}$. Repeating the above argumentation finite number of times we get $a = ae_{n-1} \leqslant e_{i-1}$.

Conversely, suppose that e and f are two n-term bases of A and $ae_i \leqslant e_{i-1}$ implies $a \leqslant e_{i-1}$ for $a \in B$, $i \geqslant 1$. We shall prove that $f \leqslant e$. What we assumed and [3, Th.2.8] together imply that A is a pseudo-complemented lattice. Then by [2 ; th.2.11] A has a chain base $f = (f_i)_{0 \leqslant i \leqslant n-1}$ not less than f and such that f_1 is the smallest dense element of the lattice A. Therefore it suffices to prove the inequality $f \leqslant e$ in the case f_1 is the smallest dense element of A. For this purpose consider monotone representations

$$e_i = a_{i1}f_1 \vee \ldots \vee a_{i,n-1}f_{n-1} \tag{4}$$

$$f_i = b_{i1}e_1 \vee \ldots \vee b_{i,n-1}e_{n-1} \tag{5}$$

$i = 1, \ldots, n-1$. By the monotonicity (4) implies $\bar{a}_{11}e_1 = 0$ whence $a_{11} = 1$.

(5) implies $\bar{b}_{11}f_1 = 0$, that is $b_{11} = 1$, f_1 being the smallest dense element of A. The first implication means $f_1 \ll e_1$, the second one means $e_1 \gg f_1$. Hence $f_1 = e_1$.

Evidently $e_1,\ldots,e_{n-1}$ and $f_1,\ldots,f_{n-1}$ are chain bases in the closed interval $[e_1, 1]$. Members of the center of this interval are of the form $a \vee e_1$, and $(a \vee e_1)e_i \leqslant e_{i-1}$ implies $a \vee e_1 \leqslant e_{i-1}$ for $i \geqslant 2$ and $a \in B$, obviously, by our hypothesis. Therefore the above argumentation may be applied to this case as well. Then in finite steps, by inductive arguments, we get the desired conclusion.

4. Proof of lemma 2. Let us firstly observe that implications

(6) $ae_i \ll e_{i-1}$ implies $ae_i = 0$ for every $a \in B$, $i = 1,\ldots,n-1$

(7) $a \vee e_{i-1} \gg e_i$ implies $a \gg e_i$ for every $a \in B$, $i = 1,\ldots,n-1$

are equivalent each to the other. In fact, the inequality $a \vee e_{i-1} \gg e_i$ for certain $i \geqslant 1$ and $a \in B$ implies $\bar{a}e_i \leqslant e_{i-1}$. then from (6) it forllows that $\bar{a}e_i = 0$, that is $a \gg e_i$. Therefore (6) implies (7). Similarly one can prove the converse statement.

Now let us consider the dual lattice A^d of A. It is easy to see (one should use the distributivity laws) that

$$1 = e_{n-1} \ll \cdots \ll e_1 \ll e_o = 0$$

is a chain base of A^d, and that $e_o \ll e_1 \ll \cdots \ll e_{n-1}$ is the least chain base in A if and only if $e_{n-1} \ll e_{n-2} \ll \cdots \ll e_o$ is the greatest one in A^d. Also the dual of (7) is the following statement

(8) $ae_{i-1} \ll e_i$ implies $a \ll e_i$ for every $a \in B$ and $i=1,\ldots,n-1$

Hence Lemma 2 holds if and only if the following statement holds: The dual A^d has the greatest n-term chain base $e_{n-1} \ll \cdots \ll e_1 \ll e_o$ if and only if $ae_{i-1} \ll e_i$ implies $a \ll e_i$ for every $a \in B$ and $i = 1,\ldots,n-1$. This is however evidently true by Lemma 1.

5. Remarks. 1. In Lemma 1 one can remove the hypothesis that $a \in B$, the center of A. It follows from the equivalence of following two implications

(9) $ae_i \ll e_{i-1}$ implies $a \ll e_{i-1}$ for every $a \in B$ and $i=1,\ldots,n-1$

(10) $xe_i \ll e_{i-1}$ implies $x \ll e_{i-1}$ for every $x \in A$ and $i=1,\ldots,n-1$

In fact, (9) follows (10) trivially. In order to show that (9) implies (10) as well let us suppose that $xe_i \leqslant e_{i-1}$ for certain $x \in A$ and that x has a monotone representation $x = \bigvee_{0 \leqslant i \leqslant n-1} x_i e_i$. Then $x_i e_i \leqslant e_{i-1}$ and so $x_i \leqslant e_{i-1}$ by (9). The monotonicity gives $x_j \leqslant e_{i-1}$ for $j \geqslant i$. Therefore $x_j e_j \leqslant e_{i-1}$ for $j > i$. The same inequality is obviously true for $j < i$. Hence $x \leqslant e_{i-1}$.

2. It follows from the above remark 1 and from [4, Lemma 3.4] that $(e_i)_{0 \leqslant i \leqslant n-1}$ is the greatest chain base of a P_o-lattice $\langle A; e_0, \ldots, e_{n-1} \rangle$ if and only if $\langle A; e_0, \ldots, e_{n-1} \rangle$ is a P_1-lattice [2, Definition 3.2].

3. Let us denote by $!x$ the greatest complemented element of a lattice A which is contained in x, [2]. It follows from (7) and Lemma 2. that if $(e_i)_{0 \leqslant i \leqslant n-1}$ is the least chain base of a P_o-lattice $\langle A; e_0, \ldots, e_{n-1} \rangle$, then $!e_i$ exists and $!e_i = 0$ for $i = 1, \ldots, e_{n-2}$.

REFERENCES

[1] G.Epstein, The lattice theory of Post algebras, TAMS, 95 (1960) 300-317.

[2] G.Epstein and A.Horn, Chain based lattices (to appear in Pac. J. of Math.).

[3] T.Traczyk, Axioms and some properties of Post algebras, Coll.Math. 10 (1963) 193-209.

[4] T.Traczyk and W.Zarębski, Generalized P_o-lattices of order (to appear in Coll.Math., 34).

[5] T.Traczyk, W.Zarębski, Post algebras are uniquely chain based lattices, (in print).

REDUCIBILITY OF POST FUNCTIONS

E. G. DuCasse

ABSTRACT

The possibility of reducing Post functions to combinations of functions having smaller domains is investigated, with emphasis being placed on reduction to Boolean functions. This reducibility property of Post functions is related to a similar property of their domains by the use of an equivalence relation.

I. INTRODUCTION

The purpose of this paper is to study how a Post function defined on an n-chain might be investigated through the use of functions defined on k-chains, where $k<n$. Ultimately, we would like to reduce the value of k to two so that the well-known properties of switching functions could be employed. It will be shown that this is not always possible.

Our investigation begins with the study of a class of equivalence relations defined on Post algebras, which include as special cases the direct products of m copies of n-chains, the domains of definition of m-variable multivalued functions, each of whose variables can assume any one of the n linearly ordered values $0<1<\ldots<n-1$. One such equivalence relation has already been shown to have important applications in the hazard-free implementation of Boolean functions [1,6]. This class of equivalence relations will be used to obtain quotient lattices of the domains of uefinition of Post functions which will in turn induce partitions in the lattices of these multivalued functions themselves. These quotients of lattices of Post functions will be seen to be isomorphic to the lattices of multivalued functions used in the reduction of the higher-order operators.

After a theory of these equivalence relations and quotient lattices has been developed, its applications to the reducibility of Post functions will be considered. A number of examples of reducible and irreducible functions are given, and theorems for determining their reducibility are presented. The paper concludes with an enumeration of reducible functions.

II. QUOTIENT LATTICES

We begin our study of quotient lattices by reproducing the definition of equivalence used to obtain hazard-free implementations of Boolean functions [1,6].

Definition 1. Let $x, y \in P$, a Post algebra [2]. We say x is equivalent

The author is with the Department of Computer and Information Science Brooklyn College, C.U.N.Y., Brooklyn, New York 11210

to y, denoted $x \sim y$, if $C_0(x) = C_0(y)$, where C_0 is the pseudo-complement defined for P.

This notion of equivalence is easily seen to be an equivalence relation, and with a little more effort can be shown to be a congruence relation. That is, it is not only reflexive, symmetric, and transitive, but is also compatible with the Post functions least upper bound, greatest lower bound, and pseudo-complement.

Theorem 2. The relation of equivalence for Post lattices is a congruence relation.

The importance of this result is that is allows us to define operations of least upper bound, greatest lower bound, and pseudo-complement for the equivalence classes which are elements of the lattice P/R, where R is the relation of equivalence. If we denote these three operations by $+_R$, $\cdot_R$, and C_0R respectively, we may define them for equivalence classes [x] and [y] as follows:

$$[x] +_R [y] = [x + y],$$

$$[x] \cdot_R [y] = [x \cdot y],$$

$$C_0R([x]) = [C_0(x)],$$

where $+$, $\cdot$, and C_0 are the least upper bound, greatest lower bound, and pseudo-complement operations, respectively, for P.

With these definitions we can establish the following result.

Theorem 3. If P is a Post algebra and R is the relation of equivalence for P, then P/R is a Post algebra under the operations $+_R$, $\cdot_R$, and C_0R.

This theorem, like Theorem 2, is a special case of a more general result to be given later.

These results summarize the most important algebraic aspects of the relation of equivalence employed in the hazard-free implementation of Boolean functions by Post functions [1,6]. It is not this relation itself, but rather a generalization of it which will be useful in the sequel. To help introduce this generalization, consider the Post algebra $[P(4)]^2$, the direct product of two 4-chains. The equivalence classes for the relation of equivalence defined on this lattice are {00}, {01, 02, 03}, {10, 20, 30}, and {11, 12, 13, 21, 22, 23, 31, 32, 33}. This last class is the interval $[e_1, U]$, where $e_1 = 11$ and $U = e_3 = 33$ is the unit element of $[P(4)]^2$. This interval is the principal filter generated by e_1, and is often denoted by $\langle e_1 \rangle$. The principal filter $\langle x \rangle$ generated by an arbitrary element $x \in L$, $(L, \leq)$ a lattice, is defined by $\langle x \rangle = \{y \in L \mid x \leq y\}$. For the relationship between filters and congruence relations, see [4].

Consider next the classes $x \cdot \langle e_1 \rangle$ defined for general Post algebras P by $x \cdot \langle e_1 \rangle = \{x \cdot y \mid e_1 \leq y\}$. The following result is straightforward.

Theorem 4. If $x \in P$, a Post algebra, then $x \cdot \langle e_1 \rangle = [x \cdot e_1, x]$.

We can employ the classes $x \cdot \langle e_1 \rangle$ in P to define a relation for Post lattices as follows.

Definition 5. Let $x, y \in P$. We define x to be e_1-equivalent to y, denoted $x \sim e_1 y$, if there exists an element $z \in P$ such that $x, y \in z \cdot \langle e_1 \rangle$.

It is not difficult to show that e_1-equivalence is an equivalence relation and that the equivalence classes of this relation are the sets $b \cdot \langle e_1 \rangle$, where $b \in P_B$, the underlying Boolean algebra or set of all complemented elements of P. In fact, the following stronger result is straightforward.

Theorem 6. e_1-equivalence is a congruence relation whose equivalence classes are the sets $b \cdot \langle e_1 \rangle$, where $b \in P_B$.

As an example, consider the lattice $[P(4)]^2$. $[P_B(4)]^2 = \{00, 03, 30, 33\}$ and $00 \cdot \langle e_1 \rangle = \{00\}$, $03 \cdot \langle e_1 \rangle = \{01, 02, 03\}$, $30 \cdot \langle e_1 \rangle = \{10, 20, 30\}$, and $33 \cdot \langle e_1 \rangle = \{11, 12, 13, 21, 22, 23, 31, 32, 33\}$. These are precisely the four equivalence classes of $[P(4)]^2$ for the equivalence relation given in Definition 1. More generally, the partition $\{b \cdot \langle e_1 \rangle \mid b \in P_B\}$ can be shown to correspond to the equivalence relation R for any Post algebra P. The classes $b \cdot \langle e_1 \rangle$ thus form the elements of the quotient lattice P/R.

We have now seen how a congruence relation and associated factor lattices can be defined for Post algebras using the filter $\langle e_1 \rangle$. We will now employ this knowledge to define more general congruence relations for Post algebras using all of the filters $\langle e_i \rangle$, $i = 1, .., n-1$.

Definition 7. Let $x, y \in P$, a Post algebra. We define x to be e_i-equivalent to y, denoted $x \sim e_i y$, if there exists an element $z \in P$ such that $x, y \in z \cdot \langle e_i \rangle$.

The following result is a generalization of Theorem 2.

Theorem 8. e_i-equivalence is a congruence relation for Post lattices.

Just as was done for the relation of equivalence, we may now define operations $+e_i$, $\cdot e_i$, and $C_0 e_i$ for the more general relation of e_i-equivalence, that is, for e_i-equivalence classes $[[x]]$ and $[[y]]$ in the quotient lattice $P/\langle e_i \rangle$. The definitions are as follows:

$$[[x]] +e_i [[y]] = [[x + y]],$$

$$[[x]] \cdot e_i [[y]] = [[x \cdot y]],$$

$$C_0 e_i([[x]]) = [[C_0(x)]].$$

These definitions enable us to give the following generalization of Theorem 3.

Theorem 9. If P is a Post algebra, then $P/\langle e_i \rangle$ is a Post algebra under

the operations $+e_i$, $\cdot e_i$, and $C_0 e_i$.

For finite Post algebras we have the following result.

Theorem 10. $[P(n)]^m/\langle e_i\rangle$ is a Post algebra isomorphic to $[P(i+1)]^m$ for all $i = 1,\ldots,n-1$.

This theorem will be important in the sequel, for it is the lattices $[P(i+1)]^m$ which will be the domains of the m-variable functions which will be used in the reduction of m-variable Post functions defined on $[P(n)]^m$. In fact, if we select the greatest lower bounds of the equivalence classes of the congruence relation $\sim e_i$ as representatives of these classes, these representatives form a lattice $[P(i+1)]^m$ themselves.

As an example of the above results, consider again the lattice $[P(4)]^2$. The filter $\langle e_2\rangle$ consists of the elements 22, 23, 32, and 33. The distinct products $z \cdot \langle e_2\rangle$ are $\{00\}$, $\{01\}$, $\{10\}$, $\{11\}$, $\{02, 03\}$, $\{12, 13\}$, $\{20, 30\}$, $\{21, 31\}$, and $\{22, 23, 32, 33\}$. These products form a Post algebra $[P(4)]^2/\langle e_2\rangle$ which is isomorphic to $[P(3)]^2$. The greatest lower bounds of these classes are, respectively, 00, 01, 10, 11, 02, 12, 20, 21, and 22. These elements themselves form a Post algebra $[P(3)]^2$.

For lattices of Post functions, Theorem 10 has an extended form. If P is the lattice $[[P(n)]^n]^m$ of m-variable Post functions, each of whose variables assumes its values from P(n), then $P/\langle e_i\rangle$ is isomorphic to a lattice of functions whose variables assume their values from the chain P(i+1). The notation $[[P(n)]^n]^m$ stands for the direct product of m copies of the direct product of n copies of P(n). This lattice contains n^{n^m} elements.

Theorem 11. $[[P(n)]^n]^m/\langle e_i\rangle$ is a Post algebra isomorphic to $[[P(i+1)]^{i+1}]^m$ for all $i = 1,\ldots, n-1$.

As an example of this theorem, consider the lattice $[[P(4)]^4]^2$ of 2-variable Post functions defined on the 4-chain P(4) and let the principal filter $\langle e_i\rangle$ be generated by e_2. The quotient lattice $[[P(4)]^4]^2/\langle e_2\rangle$ is isomorphic to $[[P(3)]^3]^2$. The importance of Theorem 11 is that the lattices $[[P(i+1)]^{i+1}]^m$ are the algebraic structures which contain the functions to which a reducible Post function from the algebra $[[P(n)]^n]^m$ can be reduced. Examples of this will be seen in the following section. Notice how this reduction of lattices of functions is related to a similar property of the domains of these functions: according to Theorem 10, $[P(4)]^2/\langle e_2\rangle$ is isomorphic to $[P(3)]^2$, the domain of the functions from the lattice $[[P(3)]^3]^2$.

With these results we are now prepared to investigate the reducibility of Post functions.

III. FUNCTION REDUCIBILITY

In this section we investigate the possibility of expressing a Post function from the lattice $[[P(n)]^n]^m$ in terms of functions from

the lattice $[[P(k)]^k]^m$ for $k<n$. We begin this investigation of reducibility with a formal definition of what it means for a Post function to be reducible.

Definition 12. Let $f: [P(n)]^m \to P(n)$ be a surjective m-variable Post function. Then f is said to be reducible with respect to $P(k)$, $2 \leq k < n$, if $f|K^m$ has image K for all ${}_nC_k$ k-element subsets K of $P(n)$.

Notice that if $m \geq n$, f is not reducible with respect to $P(k)$ for any $k<n$. This follows because any argument m-tuple $(i_1, \ldots, i_m)$ which contains all the elements of $P(n)$ cannot be an argument of any restriction $f|K^m$ of f where the cardinality of K is less than n.

Theorem 13. A necessary condition for a function $f: [P(n)]^m \to P(n)$ to be reducible with respect to $P(k)$ for any $k<n$ is that $n>m$.

In view of this theorem, we henceforth assume that $n>m$.

We stated in the Introduction that we would like to be able to reduce every reducible Post function to a set of switching functions, but that it would be shown that this is not possible. As a consequence of Theorem 13, we find that $f: [P(n)]^m \to P(n)$ cannot be reduced to a set of switching functions, that is, functions defined on a direct product of 2-chains, unless m, the number of variables, is at most two. Thus, in particular, no 3-variable Post function is reducible with respect to $P(2)$.

An argument similar to that used to establish Theorem 13 shows that even if $n>m$, f will not be reducible with respect to $P(k)$ when $k<m$. Thus we have the following result.

Theorem 14. A necessary condition for a function $f: [P(n)]^m \to P(n)$ to be reducible with respect to $P(k)$ is that $k \geq m$.

In view of this result, we henceforth assume that $k \geq m$. Thus for reducibility to be possible we must have the relationship $n>k \geq m$.

As an example of a reducible function f, consider the minimum function $MIN(x,y): [P(4)]^2 \to P(4)$ which selects the smaller of its two arguments. This function is surjective and is reducible with respect to $P(3)$. To see this we examine the restrictions $f|\{0,1,2\}^2$, $f|\{0,1,3\}^2$, $f|\{0,2,3\}^2$, and $f|\{1,2,3\}^2$. Each restriction $f|\{a,b,c\}^2$ maps its domain onto $\{a,b,c\}$. The reducibility of f with respect to $P(3)$ follows from Definition 12.

To see the relationship between this reduction and the theory of factor lattices developed in Section II, note that by Theorems 10 and 11 and Definition 12, if we are interested in reduction of a Post function with respect to $P(k)$, the reduced functions will be elements of the lattice of functions $[[P(n)]^n]^m/\langle e_{k-1}\rangle \cong [[P(k)]^k]^m$, each function having as its domain $[P(n)]^m/\langle e_{k-1}\rangle \cong [P(k)]^m$. So to obtain the factor lattice containing the reduced function $f|\{a,b,c\}^2$ we select the principal filter $\langle e_{3-1}\rangle = \langle e_2\rangle$. Forming quotient lattices of the

algebra of functions $[[P(4)]^4]^2$ and their domain $[P(4)]^2$ by the filter $\langle e_2 \rangle$, we obtain the lattice $[[P(3)]^3]^2$ of the reduced function $f|\{a,b,c\}^2$ and its domain $[P(3)]^2$.

Next consider the reducibility of f with respect to P(2). We must now examine six functions: $f|\{0,1\}^2$, $f|\{0,2\}^2$, $f|\{0,3\}^2$, $f|\{1,2\}^2$, $f|\{1,3\}^2$, and $f|\{2,3\}^2$. Each restriction $f|\{i,j\}^2$ has image $\{i,j\}$, and the reducibility of f with respect to P(2) follows. Note that if a function is reducible with respect to P(k), the reduced functions are unique. To relate the reducibility of f with respect to P(2) to the theory of factor lattices, the ideal selected to form these quotients must be $\langle e_1 \rangle$. The restriction $f|\{i,j\}^2$ is an element of the quotient lattice of functions $[[P(4)]^4]^2/\langle e_1 \rangle \cong [[P(2)]^2]^2$, whose members have domain $[P(4)]^2/\langle e_1 \rangle \cong [P(2)]^2$.

As an example of a surjective Post function which is not reducible with respect to a subchain of the chain from which its variables assume their values, consider any function g mapping $[P(3)]^2$ onto P(3) having the property that g(0,0) = 1. When we examine $g|\{0,2\}^2$ we find that g is not reducible with respect to P(2) since $1 \notin \{0,2\}$.

This example reveals a further property which a function must possess in order to be reducible, that is, it points out another necessary (but not sufficient) condition for reducibility. This condition is that for an m-variable function to be reducible, it must be the case the $f(i,\ldots,i) = i$ for all $i \in P(n)$.

Theorem 15. A necessary condition for a function $f: [P(n)]^m \to P(n)$ to be reducible with respect to P(k) is that $f(i,..,i) = i$ for all $i \in P(n)$.

If there is some $i \in P(n)$ for which $f(i,\ldots,i) = j$, $j \neq i$, the restriction of f to any k^m-element subset K^m of $[P(n)]^m$ containing the m-tuple $(i,\ldots,i)$ but for which $j \notin K$ does not even map K^m into K, let alone onto it, and the theorem is established.

To see that the necessary condition of Theorem 15 is not sufficient to guarantee that a Post function f be reducible with respect to P(k), consider the function $f: [P(3)]^2 \to P(3)$ defined by $f(i,i) = i$ for $i = 0, 1, 2$, and $f(i,j) = 0$ when $i \neq j$. Thus $f(1,2) = f(2,1) = 0$. Then f is not reducible with respect to P(2) because $f|\{1,2\}^2$ has image $\{0, 1, 2\}$.

The reason this function f is not reducible with respect to P(2) is that f(1,2) and f(2,1) are not elements of $\{1,2\}$. We can strengthen Theorem 15 by requiring that $f(i_1,\ldots,i_m) \in \{i_1,\ldots,i_m\}$ for all n^m input combinations $(i_1,\ldots,i_m)$. This requirement, of which $f(i,\ldots,i) = i$ is a special case, is also a necessary condition for f to be reducible.

Theorem 16. A necessary condition for a function $f: [P(n)]^m \to P(n)$ to be reducible with respect to P(k) is that $f(i_1,\ldots,i_m) \in \{i_1,\ldots,i_m\}$ for all n^m argument m-tuples from $[P(n)]^m$.

We give an argument independent of that used to establish Theorem 15. If this condition were not true, we could easily show that f is not

reducible with respect to $P(k)$ by selecting a set of k elements of $P(n)$ including $i_1,\ldots,i_m$, but excluding $f(i_1,\ldots,i_m)$. The absence of the closure property for f restricted to the direct product of m copies of the selected set shows that f is not reducible with respect to $P(k)$, thus establishing Theorem 16.

We have now seen a number of necessary conditions for a Post function to be reducible. None of these conditions, however, has been shown to be sufficient, and in fact we have demonstrated that the conditions established as necessary in Theorems 13, 14, and 15 are not sufficient to guarantee reducibility. We will now show that the necessary condition of Theorem 16 is also sufficient, thus providing us with a characterization of reducibility for Post functions.

Theorem 17. A Post function $f: [P(n)]^m \to P(n)$ is reducible with respect to $P(k)$ if and only if $f(i_1,\ldots,i_m) \in \{i_1,\ldots,i_m\}$ for all n^m argument m-tuples from $[P(n)]^m$.

Having shown necessity in Theorem 16, we turn now to sufficiency. Let K be a k-element subset of $P(n)$. The condition $f(i_1,\ldots,i_m) \in \{i_1,\ldots,i_m\}$ guarantees that the image of $f|K^m$ is a subset of K. To see that $f|K^m$ is surjective, recall that $f(i,\ldots i) = i$ is a consequence of the condition $f(i_1,\ldots,i_m) \in \{i_1,\ldots,i_m\}$. Since $(i,\ldots,i) \in K^m$ for all $i \in K$, it follows that every $i \in K$ is the image of an argument of $f|K^m$. In fact, this argument is $(i,\ldots,i)$. This concludes the proof.

Now suppose a function $f: [P(n)]^m \to P(n)$ is reducible with respect to $P(k)$, where $k > 2$. Then f is also reducible with respect to $P(k-1)$. For if it were not, then according to Theorem 17, there would exist a set L of $k-1$ elements of $P(n)$ for which $f|L^m$ would not have codomain L. There are two cases to consider. Either there is an element x in the image of $f|L^m$ which is not in L or there is an element y of L which is not in the image of $f|L^m$. In the first case, the existence of an element x in the image of $f|L^m$ which is not in the $(k-1)$-element set L implies that there is at least one k-element subset K of $P(n)$ such that $x \notin K$, but for which $f|K^m$ has image containing x. This contradicts the assumed reducibility of f with respect to $P(k)$.

Now consider the case where L contains an element y which is not in the image of $f|L^m$. Then $(f|L^m)(y,\ldots,y) \neq y$. Hence, $(f|K^m)(y,\ldots,y) \neq y$ for any k-element subset K of $P(n)$. According to Theorem 15, f is not reducible with respect to $P(k)$, a contradiction. This establishes our claim that a function reducible with respect to $P(k)$ is also reducible with respect to $P(k-1)$. It follows by induction that a Post function $f: [P(n)]^m \to P(n)$ which is reducible with respect to $P(n-1)$ is reducible with respect to $P(k)$ for all k satisfying $m \leq k < n$.

Theorem 18. If $f: [P(n)]^m \to P(n)$ is reducible with respect to $P(k)$, then f is reducible with respect to $P(j)$ for all j satisfying $m \leq j \leq k$.

Having shown that reducibility with respect to $P(k)$ implies reducibility with respect to $P(k-1)$, we next prove the converse. Assume f if reducible with respect to $P(k)$ for $k < n-1$. if f is not reducible with respect to $P(k+1)$, we again have two cases to consider. There must

exist a $(k+1)$-element subset M of $P(n)$ such that either there is an element x in the image of $f|M^m$ which is not in M or there is an element y of M which is not in the image of $f|M^m$. In the first case, there must exist at least one k-element subset K of $P(n)$ such that $x \notin K$, while x is a member of the image of $f|K^m$. This contradicts the reducibility of f with respect to $P(k)$.

In the second case, we have $(f|M^m)(y,\ldots,y) \neq y$. Hence, $(f|K^m)(y,\ldots,y) \neq y$ for every k-element subset K of $P(n)$ containing y. By Theorem 15, f is not reducible with respect to $P(k)$, contradicting our assumption. Thus a function reducible with respect to $P(k)$, $k<n-1$, is also reducible with respect to $P(k+1)$. It follows by induction that a function reducible with respect to $P(k)$ is reducible with respect to $P(j)$ for all $k \leq j \leq n-1$.

Theorem 19. If $f: [P(n)]^m \to P(n)$ is reducible with respect to $P(k)$, $k<n-1$, then f is reducible with respect to $P(j)$ for all j satisfying $k \leq j < n$.

We have established the following result also.

Theorem 20. A Post function f is reducible with respect to $P(k)$ if and only if it is reducible with respect to $P(k+1)$ for all k satisfying $m \leq k < n-1$.

The next result is a consequence of Theorems 18 and 19.

Theorem 21. If a Post function is reducible with respect to $P(k)$, then it is reducible with respect to $P(j)$ for all j such that $m \leq j < n$.

In the light of this result, we shall henceforth refer to a Post function as being reducible or irreducible without reference to a particular chain. Notice that the transitivity of reducibility is a special case of Theorem 21.

As an example illustrating the advantages of using Theorems 17 and 21 instead of Definition 12 in determining the reducibility of Post functions, consider the maximum function $MAX(x,y): [P(5)]^2 \to P(5)$ which has as its value the larger of its two arguments. Since $MAX(x,y) \in \{x,y\}$, it follows from Theorem 17 that this function is reducible. According to Theorem 21, MAX is reducible with respect to $P(2)$, $P(3)$, and $P(4)$. The restrictions of the MAX function to each of these chains are again MAX functions. If we were to verify the reducibility of this function with respect to $P(2)$, $P(3)$, and $P(4)$ by employing Definition 12, we would have to verify the surjective closure property of this definition for ${}_5C_4 + {}_5C_3 + {}_5C_2 = 25$ functions. The savings in time and effort gained by employing Theorems 17 and 21 is obvious. The factor lattices containing the reduced functions for $P(2)$, $P(3)$, and $P(4)$ are $[[P(5)]^5]^2/\langle e_1 \rangle$, $[[P(5)]^5]^2/\langle e_2 \rangle$, and $[[P(5)]^5]^2/\langle e_3 \rangle$ respectively.

Having established a number of results concerning the reducibility of multivalued functions, we turn now to the problem of determining the number of reducible elements in a lattice of Post functions $[[P(n)]^n]^m$.

The number of m-variable functions defined on an n-chain is n^{n^m}. We will soon see that the number of these functions which are reducible is generally relatively small compared to the total number. For example, there are $3^{3^2} = 19{,}683$ 2-variable Post functions defined on the 3-chain, while the number of such functions which are reducible will be shown presently to be only 32.

To see how the number of reducible functions can be calculated, we consider a set of numbers called Stirling numbers of the second kind. For $n \geq m$, the Stirling number S_n^m of the second kind is the number of partitions of a set of n objects into m classes. The numbers S_n^m can be calculated recursively from the following recursion relations:

$$S_{n+1}^k = S_n^{k-1} + kS_n^k, \qquad \text{where } 1 < k < n,$$

$$S_n^1 = S_n^n + 1.$$

Before determining how many of the n^{n^m} m-variable functions are reducible, we determine how many satisfy the first condition of Definition 12, that is, we determine how many are surjective.

Theorem 22. The number of surjective functions $f: [P(n)]^m \to P(n)$ is $n!S_{nm}^n$.

To see this, first notice that each surjection of $[P(n)]^m$ onto $P(n)$ induces a partition of $[P(n)]^m$ into n distinct classes. Conversely, each partition of $[P(n)]^m$ into n classes corresponds to n! surjections of $[P(n)]^m$ onto $P(n)$.

We turn now to the determination of the number of reducible functions among the total of n^{n^m}. Let $T_{m,n}^k$ denote the number of argument m-tuples $(i_1,\ldots,i_m)$ of a function $f: [P(n)]^m \to P(n)$ which have exactly k distinct i_j. We have the following result.

Theorem 23. $T_{m,n}^k = {}_nP_k \cdot S_m^k$, where ${}_nP_k$ is the number of permutations of n objects taken k at a time.

We conclude this section with an expression for the number of reducible m-variable Post functions, all of whose variables assume their values from the chain P(n). Let R_n^m denote the number of reducible surjections $f: [P(n)]^m \to P(n)$. The following theorem gives a formula for determining R_n^m for $n \geq m$.

Theorem 24. $R_n^m = \prod_{i=1}^{m} iT_{m,n}^i$.

As an application of this result, we calculate the number of reducible 2-variable Post functions defined on $[P(3)]^2$. We have $R_3^2 = \prod_{i=1}^{2} iT_{2,3}^i = 1 \cdot 2^6 = 32$, as mentioned previously. Using Theorem 17 it is easy, but tedious to write down these 32 functions explicitly.

IV. CONCLUSION

A theory of factor lattices of Post algebras based on principal filters generated by the elements $e_1,\ldots,e_{n-1}$ has been developed and

related to a concept of reducibility for Post functions. A characterization of reducibility has been established and an enumeration of reducible functions has been given. Relating Post functions to simpler multivalued operators defined on smaller lattices should aid in the determination of properties of the higher-order functions.

REFERENCES

1. DuCasse, E. and Metze, G., "Hazard-Free Realizations of Boolean Functions Using Post Functions," Conference Record of the 1973 International Symposium on Multiple-Valued Logic Design, Toronto, Canada, pp. 59-67, May, 1973.

2. Epstein, G., "The Lattice Theory of Post Algebras,: Transactions of the American Mathematical Society, Vol. 95, pp. 300-317, 1960.

3. Post, E., "Introduction to a General Theory of Elementary Propositions," American Journal of Mathematics, Vol. 43, pp. 163-185, 1921.

4. Rasiowa, H. and Sikorski, R., Mathematics of Metamathematics, Monografie Matematyczne, Tom 41, Polska Akademia Nauk, Warzawa, 1963.

5. Rosenbloom, P., "Post Algebras, I. Postulates and General Theory," American Journal of Mathematics, Vol. 64, pp. 167-188, 1942.

6. Wojcik, A., "Relationships Between Post and Boolean Algebras with Application to Multivalued Switching Theory," Coordinated Science Laboratory Report R-512, University of Illinois, Urbana, 1971.

SUMMARY

RECENT DEVELOPMENTS IN THE THEORY OF POST ALGEBRAS†

Philip Dwinger
University of Illinois at Chicago Circle
USA

The development of universal algebra and category theory has for a great deal determined the direction into which the study of Post algebras and its generalizations has developed in recent years. In this summary some examples will be presented to illustrate this trend.

It is well known [5] that if L is a Post algebra of order n, then the chain of constants is uniquely determined. Equivalently, in the equational class of Post algebras of order n, the nullary operations are uniquely determined. It is also known [9] that a Post algebra of order n can be defined as the coproduct in the category D_{01} of distributive lattices with 0 and 1, of a Boolean algebra and a chain of length n This leads to a notion of generalized Post algebra: $L = B*C$, where B is a Boolean algebra, C is a chain with 0 and 1 and where $*$ means coproduct in D_{01}. The question then arises whether B and C are uniquely determined. That B is uniquely determined is an easy consequence from the fact that B is the center of L. But unlike in the finite case, C in general is not uniquely determined. It has been shown [1] that C is uniquely determined, provided C is rigid, i.e., C does not admit a non-trivial order isomorphism. On the other hand, if C is a chain with 0 and 1 and C is not rigid and $L = B*C$, where $B \neq 2$, then there exists a chain C' (also with 0 and 1) such that $L = B*C'$ and $C' \neq C$ [1]. Recently a more general problem has been considered. Suppose L_1, L_2, L_2' are objects of D_{01} and $L_1 * L_2 = L_1 * L_2'$. What can be said about L_2 and L_2'? It has been proven [4], [10] using algebraic methods that always L_2 and L_2' are isomorphic, but not necessarily $L_2 = L_2'$. However, if L_2 is rigid then $L_2 = L_2'$. A converse of this result, similar to the result for the special case stated above, also holds. It is rather surprising that these and other results can also be proved topologically, using the coequivalence between the category D_{01} and the category of bounded Stone spaces and strongly continuous maps, and the fact that that topological product and direct product in the last category are the same [8]. Whereas the algebraic proofs of the aforementioned results are rather complicated the topological proofs then become rather simple exercises in point set topology.

Post algebras of order n are also relative Stone algebras. An interesting generalization which is related to this property is the class of Stone algebras of order n and which has been investigated in [7].

A Post algebra of order n can also be considered as the lattice of continuous functions of a Boolean space to the n-elements discrete chain [6]. A generalization to infinite discrete chains has been considered in

† This paper will appear in the book Modern Uses of Multiple-Valued Logic.

[3]. Let $L = \langle X, C \rangle$ be the lattice of all continuous functions on a Boolean space to an arbitrary discrete chain C (thus C does not need to have a 0 or a 1). It has been proved in [2] that again L determines X up to homeomorphisms and C up to isomorphisms. Furthermore it was proved in [2] that every automorphism of L maps the set of constant functions onto itself if and only if C is rigid. This strongly generalizes the results mentioned in the first part of this abstract.

REFERENCES

[1] R. Balbes and Ph. Dwinger, Distributive Lattices, University of Missouri Press, 1974.

[2] W.J. Blok, Generalized Post algebras, Masters Thesis, University of Amsterdam, 1972.

[3] C.C. Chang and A. Horn, Prime ideal characterization of generalized Post algebras, Proceedings of the Symposium of Pure Math., AMS, 2(1961), 43-48.

[4] S.D. Comer and Ph. Dwinger, Cancellation for bounded distributive lattives, Notices of the AMS, Abstract 711-06-10, 21(1974), A-44.

[5] Ph. Dwinger, Notes on Post algebras I and II, Indag. Math. 28(1966)-462-478.

[6] G. Epstein, The lattice theory of Post algebras, Trans. Amer. Math. Soc. 95(1960), 300-317.

[7] T. Katrinak and A. Mitschke, Stonesche Verbande der Ordung n und Postalgebren, Math. Ann., (1972), 13-20.

[8] A. Nerode, Some Stone spaces and recursion theory, Duke Math. J., (1959), 397-406.

[9] G. Rouseau, Post algebras and pseudo-Post algebras, Fund. Math. 67 (1970), 133-145.

[10] J. Strecker, The strong cancellation property for the free product of bounded distributive lattices, Thesis-Vanderbilt University, 1974.

SOME FURTHER PROPERTIES OF THE Pi-LOGICS

Vaclay Pinkava
Severalls Hospital
Colchester, Essex, U. K.

Summary. After repeating briefly the previous main results concerning the Pi-logics, some further complete subclasses are considered, mainly such consisting of the cyclic negation and only one of the relevant two argument types of functions.

Introduction. Some time ago the present author discovered a fairly general class of functionally complete sets of functions in finite multiple-valued logics. Most, if not all of the current multi-valued logics, like that of Post [3], Rosser and Turquette [5], Zhegalkin [6], Aizenberg and Rabinovich [4], are actually instances of this general class of functions. If the isomorphic systems were included as well, the class would appear still more general.

In 1974 Kohout [1] presented a paper dealing with the system mainly from the point of view of applications in computing and gave also a partial classification of the system. When doing so he christened the system with the name of its discoverer. Because of this, instead of talking about "a system of logics" or the like, but avoiding being immodest, I am referring to the particular class as the Pi-logics. The original article, as yet unpublished, has been submitted to the IEEETC for publication and/or storage in their Repository, wherefrom it should be available [2]. For questions connected with completeness see [1]. The paper quoted gives also the most important Theorems in an Appendix. The proofs and more details can be found in [2].

Before dealing with the topic of the present paper I consider it desirable to repeat once again the basic results contained in [2] and quoted in [1]. Here and throughout this paper the following formalisms will be used: Logical variables will be symbolised by: $v_1, v_2, \dots, v_n, v_i, v$. If $f(v_1, v_2)$ be a function and if a property in question may depend on either v_1 or v_2, then we shall write v_i, v_j rather than v_1, v_2 . In that case $i, j = 1, 2$ and if $i = 1$ then $j = 2$ and vice versa. E.g., instead of writing $f(0, v_2) = f(v_1, 0) = 0$, we may write just: $f(0, v_j)$ or $f(v_i, 0) = 0$. If, e.g., $\bigtriangledown$ be a symbol meaning a connective or functor like: $v_1 \bigtriangledown v_2$, then $\bigtriangledown_{v_i=1}^{v_i=n} v_i$ will have the meaning of $v_1 \bigtriangledown v_2 \bigtriangledown v_3 \bigtriangledown \dots \bigtriangledown v_n$. Here $\bigtriangledown$ and $\bigtriangledown$ has the meaning of a variable assuming values from the set of symbols meaning two-argument functors.

The cyclic negation: $v + 1 \pmod k = \tilde{v}$. Then: $\tilde{v}^{\chi}$ means the number superpositions of $\sim$ over v . Owing to the definition it is: $\tilde{v}^k = \tilde{v}^0 = v$; $\tilde{\tilde{v}}^n_m = \tilde{\tilde{v}}^m_n = \tilde{v}^{m+n \pmod k}$. An expression of the general type: $\bigtriangledown \{\tilde{v}^{\chi}\}$ means a repeated operation $\bigtriangledown$ over the whole set of functions: $\{\tilde{v}^0, \tilde{v}^1, \tilde{v}^2, \dots, \tilde{v}^{k-1}\}$. An expression of the type: $\bigtriangledown [\{\tilde{v}^{\chi}\} - \tilde{v}^{\delta}\}$ means repeated operation of $\bigtriangledown$ over the set $\{v^{\chi}\}$, save for one function $\tilde{v}^{\delta}$, i.e., $\tilde{v}^0 \bigtriangledown \tilde{v}^1 \bigtriangledown \tilde{v}^2 \bigtriangledown \dots \bigtriangledown \tilde{v}^{\delta-1} \bigtriangledown \tilde{v}^{\delta+1} \bigtriangledown \dots \bigtriangledown \tilde{v}^{k-1}$. $(\delta = 0, 1, 2, \dots, k-1)$.

The main results of [2] are as follows: There are three incompletely defined functions or types of functions for any finite multi-valued logic /$k \geq 2$/ :

a) $$v_1 \diamondplus v_2 = \begin{cases} 0 & \text{iff } v_i = 0 \\ 1 & \text{if } v_i \neq 0 \text{ , and } v_j = 1 \end{cases} ;$$

b) $$v_1 \boxplus v_2 = \begin{cases} 0 & \text{if } v_i = 0 \\ v_i & \text{if } v_i \neq 0 \text{ and } v_j = 1 \end{cases} ;$$

c) $$v_1 \odot v_2 = v_j \text{ if } v_i = 0 \quad .$$

Any set of functions of the above given general type plus the cyclic negation $\tilde{v} = v+1 \pmod k$, i.e., any set: $\{ \diamondplus , \boxplus , \odot , \sim \}$ is functionally complete, i.e., capable of expressing any function of the respective k-valued logic. /Theorem 1/

This follows from Lemma 3, which proves that any k-valued function can be expressed by means of a formula of the general type:

$$f(v_1, v_2, \dots, v_n) = \bigodot_{\lambda=1}^{\lambda=m\leq k^n} [c_\varkappa \boxplus (\bigboxplus_{\ell=1}^{\ell=n} \psi_{a_\ell}(v_\ell))]_\lambda$$

which is actually a very generalized analogon of the complete normal disjunctive form in binary logic. Usually we are concerned only with functions $f(v_1, v_2, \dots, v_n) \neq 0$, and in that case the above formula may be changed in that the constants $c_\varkappa$ are replaced by c_γ, $\gamma = 1, 2, \dots, k-1$. The type of formula given above employs the connectives /functors/ : $\odot$, $\boxplus$, $\diamondplus$, constants $c_\varkappa$, $(\varkappa = 0, 1, \dots, k-1)$ or c_γ, $(\gamma = 1, 2, \dots, k-1)$ and characteristic functions of the general type:

$$\psi_\varkappa(v) = \begin{cases} 1 & \text{iff } v = \varkappa \\ 0 & \text{iff } v \neq \varkappa \end{cases} .$$

It may be shown that:

$$c_\varkappa = \overbrace{\diamondplus \{\tilde{v}^\varkappa\}}^{\varkappa} \qquad \text{/Lemma 1/}$$

and

$$\psi_\varkappa(v) = \diamondplus [\{\tilde{v}^\varkappa\} - \tilde{v}^{k-\varkappa \pmod k}] \quad , \qquad \text{/Lemma 2/}$$

so that the set $\{ \diamondplus , \boxplus , \odot , \sim \}$ is complete.

The connective '$\diamondplus$' is necessary only for generating constants and characteristic functions. With the values 1, 0, the functor '$\boxplus$' behaves the same as '$\diamondplus$' . From this it follows:

$$f(v_1, v_2, \dots, v_n) = \bigodot_{\substack{\lambda=1 \\ f(a_1,a_2,\dots,a_n)\neq 0}}^{\lambda=m\geq k^n} [c_\gamma \boxplus (\bigdiamondplus_{\ell=1}^{\ell=n} \psi_{a_\ell}(v_\ell))]_\lambda = \bigodot_{\substack{\lambda=1 \\ f(a_1,a_2,\dots,a_n)\neq 0}}^{\lambda=m\geq k^n} [c_\gamma \boxplus (\bigdiamondplus_{\ell=1}^{\ell=n} \lambda_{a_\ell}(v_\ell))]_\lambda$$

It may be realized that the complete normal disjunctive form in binary logic is actually a special instance of the above type of formula: The '$\boxplus$' and '$\diamondplus$'-type are completely defined for $k = 2$ and coincide in $v_1 \& v_2$ or 'AND' . The $\odot$-type allows two different complementations, thus generating either the ordinary OR or the exclusive OR

/non-equivalence/ . The characteristic functions are represented by negations or their absence above variables and every conjunct can be viewed as multiplied by a constant $c_\gamma = 1$.

For every functor $\square \in \{ \boxplus \}$ there exists a functor $\bigcirc \in \{\odot\}$ such that:

$$\overbrace{v_1^{\sim 1} \square v_2^{\sim 1}}^{k-1} = v_1 \bigcirc v_2 \quad , \quad \overbrace{v_1^{\sim k-1} \bigcirc v_2^{\sim k-1}}^{1} = v_1 \square v_2 . \text{ /Theorem 2/}$$

Thus, any set of the type $\{ + , + , \sim \}$ is also functionally complete. /Theorem 3/

However, not every set of the type $\{ \diamond , \odot , \sim \}$ will be complete. This applies only to functors of a subclass: $\{\textcircled{\vee}\} \subset \{\odot\}$ such that: $v_i \textcircled{\vee} (k-1) = k-1$ at the same time. Then also always:

$$\overbrace{\tilde{v}_1^{k-1} \textcircled{\vee} \tilde{v}_2^{k1}}^{1} = v_1 \boxplus v_2$$

and $\{\textcircled{\vee}, \diamond , \sim \}$ is complete. /Theorem 4/

It may be observed that $\max(v_1, v_2)$ is a $\textcircled{\vee}$-type functor. Together with $\min(v_1 , v_2) \in \{ \diamond \}$ and the cyclic negation it forms the familiar Post system.

If $v_1 \diamond v_2$ is such that $\diamond \in \{ \diamond \}$ and $v_i \diamond (k-1) = v_i$, then $\{\diamond , \boxplus , \sim\}$ is also complete, as:

$$\overbrace{\tilde{v}_1^{k-1} \diamond \tilde{v}_2^{k-1}}^{1} = v_1 \odot v_2 \quad . \text{ /Theorem 5/}$$

Conversely, any set of the type $\{ \boxplus , \textcircled{\wedge} , \sim\}$ will be complete, where $\textcircled{\wedge} \in \{\odot\}$ and $v_i \textcircled{\wedge} 1 = 1$, $v_i \textcircled{\wedge} 2 = 2$ (for $v_i \neq 1$) , at the same time as:

$$\overbrace{\tilde{v}_1^{1} \textcircled{\wedge} \tilde{v}_2^{1}}^{k-1} = v_1 \diamond v_2 \quad . \text{ /Theorem 6/}$$

The system of Aizanberg and Rabinovich consisting of cyclic negation and two functions:

$$\zeta(v_1, v_2) = \begin{cases} v_i & \text{if } v_j = 1 \\ 0 & \text{otherwise} \end{cases} \quad , \quad \mu(v_1, v_2) = \begin{cases} v_i & \text{if } v_j = 0 \\ \min(v_1, v_2) & \text{otherwise} \end{cases}$$

is a $\{ \boxplus , \textcircled{\wedge} , \sim\}$-type one, as: $\zeta(v_1, v_2) \in \{ \boxplus \}$, $\mu(v_1, v_2) \in \{\textcircled{\wedge}\}$.

So far the main previous results.

Complete Pi-Systems Consisting of Just One of the Binary Operations and the Cyclic Negation. In this and the following section Theorems and Lemmata will be numbered in succession, and there will be numbers in parentheses. These latter ones refer to numbering in continuation of that given in [2].

Let us consider a function $\circledcirc \in \{\odot\}$ /representing a class of functions: $\{\circledcirc\} \subset \{\odot\}$/ such that at the same time it holds also:

$$v_1 \circledcirc v_2 = \begin{cases} v_j & \text{if } v_i = 0 \\ k-2 & \text{if } v_i = k-2 \text{ and } v_j \neq k-1 \\ k-1 & \text{iff } v_i = k-1 \end{cases}$$

Owing to the last property of $\odot \in \{\odot\}$, it will be able to generate a $\boxplus$-type, owing to the relation:

$$\tilde{v}_1^{k-1} \odot \tilde{v}_2^{k-1^1} = v_1 \boxdot v_2 \quad ; \quad \boxdot \in \{\boxplus\} \ . \qquad \text{/see Theorem 3/}$$

It will be observed that apart from being $\boxdot \in \{\boxplus\}$, the $\boxdot$ has also the following property:

$$v_i \boxdot k-1 = k-1 \text{ for any } v_i \neq 0 \ .$$

Theorem 1 /8/ The set $\{\odot, \sim\}$ is functionally complete.

Requirements for Proof: The set $\{\odot, \sim\}$ enables forming a $\boxplus$-type, $\boxdot \in \{\boxplus\}$, as shown. As it holds that:

$$f(v_1, v_2, \dots, v_n) = \bigodot_{\substack{\lambda=1 \\ f(a_1,a_2,\dots,a_n)\neq 0}}^{\lambda=m\leq k^n} [c_\gamma \boxplus (\boxplus_{\ell=1}^{\ell=n} \psi_{a_\ell}(v_\ell))]_\lambda$$

and: $\odot \in \{\odot\}$, $\boxdot \in \{\boxplus\}$, it will be sufficient and necessary to prove that the set: $\{\odot, \boxdot, \sim\}$ can generate the constants c_χ and the characteristic functions of the type: $\psi_\chi(v)$. In order to achieve this we first formulate and prove 3 Lemmata:

Lemma 1 /4/

$$c_\chi = \widetilde{\boxdot\{\tilde{v}^\chi\}} \quad ; \quad (\chi = 0, 1, 2, \dots, k-1).$$

Proof: As $\boxdot \in \{\boxplus\}$, then $v_i \boxdot 0 = 0$. Thus: $\boxdot\{\tilde{v}^\chi\} = 0$, as amongst the functions $\tilde{v}^\chi$, $(\chi = 0, 1, 2, \dots, k-1)$ there will be always one for which, for a given substitution a it will be: $\tilde{a}^\chi = 0$. Thus, disregarding the substitution, the expression $\boxdot\{\tilde{v}^\chi\}$ will be always reducible to $b \boxdot 0$ or $0 \boxdot b$, where: $b \in \{0, 1, 2, \dots, k-1\}$, so that $\boxdot\{\tilde{v}^\chi\} = 0$. However: $\tilde{0}^\chi = \chi = c_\chi$. Thus: $c_\chi = \widetilde{\boxdot\{\tilde{v}^\chi\}}$, Q. E. D.

Let us further introduce the following type of characteristic function

$$f_\chi(v) = \begin{cases} k-1 & \text{iff } v = \chi \\ 0 & \text{iff } v \neq \chi \end{cases}$$

Lemma 2 /5/

$$f_\chi(v) = \boxdot[\{\tilde{v}^\chi\} - \tilde{v}^{k-\chi(\mathrm{mod}\,k)}] \ .$$

Proof: It is: $v_i \boxdot 0 = 0$ for every v_i, and $v_i \boxdot (k-1) = k = 1$ for every $v_i \neq 0$. Thus, a function of the general type: $\boxdot[\{\tilde{v}^\chi\} - \tilde{v}^\delta]$ will assume the value of 0 for all the value substitutions, save for that value a for which $\tilde{a}^\delta = 0$. In that case /as the function $\tilde{v}^\delta$ has been removed from $\{v^\chi\}$/, the function $\boxdot[\{\tilde{v}^\chi\} - \tilde{v}^\delta] = k-1$, for always one of the remaining functions $\tilde{v}^i$ will assume that value $k-1$.

Now, the funct for which a given $\tilde{x}^\delta = 0$, will be found to be: $\tilde{v}^{k-\chi(\mathrm{mod}\,k)}$. It is namely: $\tilde{0}^\chi = \chi$, $\tilde{0}^{k-1} = k-1$, hence $0^k = \tilde{0}^0 = 0$ and hence $\tilde{x}^{k-\chi(\mathrm{mod}\,k)} = 0$. Thus:

$$f_\chi(v) = \boxdot[\{\tilde{v}^\chi\} - \tilde{v}^{k-\chi(\mathrm{mod}\,k)}] \ . \quad \text{Q. E. D.}$$

Let us introduce the function:

$$\widetilde{f_\chi(v)}^1 = \begin{cases} 0 & \text{iff } v = \chi \\ 1 & \text{iff } v \neq \chi \end{cases} \ .$$

Realising that:

$$\psi_i(v) = \begin{cases} 1 & \text{iff } v = i \\ 0 & \text{iff } v \neq i \end{cases} \qquad (i = 0, 1, 2, \ldots, k-1) \quad ,$$

we may formulate

Lemma 3 /6/:

$$\psi_i(v) = \boxed{\circ}[\{\tilde{f}^1_\chi(v)\} - \tilde{f}^1_i(v)] \quad .$$

(Here, the expression means the repeated operation of $\boxed{\circ}$ over the set of functions $\tilde{f}^1_\chi(v)$, $(\chi = 0, 1, 2, \ldots, k-1)$, save for one function $\tilde{f}^1_i(v)$, $(i = 0, 1, 2, \ldots, k-1)$.)

Proof: The characteristic function $f_i(v)$ assumes the value of $k-1$ for the argument value $v = i$ and the value of 0 for all remaining argument values. The function: $\tilde{f}^1_i(v)$ thus assumes the value of 0 for $v = i$ and the value of 1 for all $v \neq i$. When feeding the value i into the expression: $\boxed{\circ}[\{\tilde{f}^1_\chi(v)\} - \tilde{f}^1_i(v)]$, this expression will turn into 1 . For, the function $\tilde{f}^1_i(v)$, for which it would assume the value 0 has been removed and all the remaining functions will assume the value of 1 .

On the other hand, if feeding any value a , such as $a \neq i$ into the above expression, the expression will assume the value of 0 . For, there will always be one function $\tilde{f}^1_j(v)$ for which $\tilde{f}^1_j(a) = 0$, which will be sufficient to turn the whole expression into 0 . Thus:

$$\psi_i(v) = \boxed{\circ}[\{\tilde{f}^1_\chi(v) - \tilde{f}^1_i(v)] \quad . \qquad \text{Q. E. D.}$$

Proof of Theorem 1 /8/ From the Lemmata 1 /4/, 2 /5/, 3 /6/, it follows that the constants $\{c_\gamma\}$; $(\{c_\gamma\} \subset \{c_\chi\})$, and the characteristic functions of the type $\psi_\chi(v)$ can be generated by the set: $\{\boxed{\circ}, \sim\}$. '$\boxed{\circ}$' in its turn can be generated by the set $\{\odot, \sim\}$. Thus, the set $\{\odot, \sim\}$ is functionally complete. Q. E. D.

Theorem 2 /9/ The set $\{\boxed{\circ}, \sim\}$, (where $\boxed{\circ} \in \{\boxplus\}$ and $v_i \boxed{\circ} k-1 = k-1$ for any $v_i \neq 0$, is functionally complete.

Proof: $\{\boxed{\circ}, \sim\}$ can generate constants and characteristic functions as indicated in the proof to Theorem 1 /8/. Any set $\{\boxplus, \sim\}$ can generate a $\odot$-type. Thus, also a set $\{\boxed{\circ}, \sim\}$. (Where, incidentally, the $\odot$-type will be of the subclass: $\{\circledcirc\}$.) Thus, the set is functionally complete. Q. E. D.

Let us now introduce the following type of characteristic function:

$$\zeta^a_i(v) = \begin{cases} a & \text{iff } v = i \\ 0 & \text{iff } v \neq i \end{cases} \quad ,$$

where: $a = 0, 1, 2, , \ldots, k-1$; $i = 0, 1, 2, , \ldots, k-1$. It is further obvious that we may write: $\zeta^a_i(v) = a \boxplus \psi_i(v)$. Let us now introduce the following one-argument function:

$$F(v) = \bar{v} = \bigodot_{\substack{b=0 \\ F(v)\neq 0}}^{b=k-2} \zeta^b_a(v) = \bigodot_{b=0}^{b=k-2} [a \boxplus \psi_b(v)] \quad ,$$

where: $a, b = 0, 1, 2, , \ldots, k-1$ and: $a + b = k-1$. /According to the condition: $F(v) \neq 0$, the expression $\zeta^0_{k-1}(v)$ is dropped./ It can be shown that it holds: $\bar{\bar{v}} = v$.

Lemma 4 /7/

a) $\overline{\overline{v}_1 \odot \overline{v}_2} = v_1 \diamond v_2$; $(\diamond \in \{\oplus\})$

b) $\overline{\overline{v}_1 \diamond \overline{v}_2} = v_1 \odot v_2$.

Proof:

a) As: $v_1 \odot v_2 = \begin{cases} v_j & \text{if } v_i = 0 \ , \\ k-2 & \text{if } v_i = k-2 \text{ and } v_j \neq k-1 \ , \\ k-1 & \text{if } v_i = k-1 \ , \end{cases}$

we have: $\overline{0 \odot \overline{v}_i} = \overline{k\text{-}1 \odot \overline{v}_i} = \overline{k\text{-}1} = 0$, $\overline{1 \odot \overline{v}_i} = \overline{k\text{-}2 \odot \overline{v}_i} = \overline{k\text{-}2} = 1$ (for $v_i \neq 0, k-1$) , $\overline{k\text{-}1 \odot \overline{v}_i} = \overline{0 \odot \overline{v}_i} = v_i$. Thus, we have: $f(0, v_i) = f(v_i, 0) = 0$, $f(1, v_i) = f(v_i, 1) = 1$, which are the characteristics of a $\oplus$-type functor. In addition, it holds also:

$$f(k-1, v_i) = v_i$$

which adds a further characteristic, defining a subclass $\{\diamond\} \subset \{\oplus\}$.

b) As: $\overline{v} = v$ and $\overline{\overline{v}_1 \odot \overline{v}_2} = v_1 \diamond v_2$, it follows that:

$$\overline{\overline{\overline{v}_1 \odot \overline{v}_2}} = \overline{\overline{v}_1 \diamond \overline{v}_2} = v_1 \odot v_2 \ .$$

The Lemma is proved.

Corrolary Whilst $\{\odot, \sim\}$ and $\{\boxdot, \sim\}$ are complete sets, this does not generally apply to $\{\diamond, \sim\}$. For, although $\overline{\overline{v}_1 \diamond \overline{v}_2} = v_1 \odot v_2$ the function $\overline{v}$ *cannot* be generated from the set $\{\diamond, \sim\}$ alone without further specifications: The functor '$\diamond$' is incidentally identical with the $\oplus$-type dealt with previously and allowing generation of a '$\odot$' of the subclass '$\wedge$' via: $\widetilde{v_1^{k-1}} \cdot \widetilde{v_2^{k-1}}^{1} = v_1 \wedge v_2$. However, in order to complete the set $\{\diamond, \wedge, \sim\}$ must allow generating a $\boxplus$-type as well. But, in order to be convertible into a $\boxplus$-type, a $\odot$-type must have the property of: $v_i \odot (k-1) = k-1$, which the $\wedge$-type is evidently lacking.

The Post functor $v_1 \vee v_2 = \max(v_1, v_2)$ is a $\odot$-type and thus suffices, together with the cyclic negation, to form a complete system. On the other hand, the Post functor $v_1 \,\&\, v_2$ does not form a complete system with cyclic negation for any $k > 2$. For $k = 2$ it coincides with the $\boxdot$-type, and thus forms a complete system with negation. /For $k = 2$ the '$\sim$' and the '$-$' negation again coincide ./

The system $\{\triangle, \sim\}$ will be complete, the function $v_1 \triangle v_2$ being defined as follows:

$$v_1 \triangle v_2 = \begin{cases} 0 & \text{if } v_i = 0 \\ \max(v_1, v_2) & \text{otherwise} \end{cases} \ .$$

It is actually the corresponding $\boxdot$-type to $\max(v_1, v_2)$ as $\odot$-type, and it holds:

$$\widetilde{v_1^{1}} \triangle \widetilde{v_2^{1}}^{k-1} = \max(v_1, v_2) \ .$$

The generality of the Pi-logics can be increased considerably, if we introduce all the possible relevant morphisms with respect to values and functions: This will be dealt with in some detail in another paper.

For instance, let us consider a completely ordered set of any kind of symbols $\{\alpha_i\}$, so that $\{\alpha_i\} = \{\alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_{k-1}\}$. Let v be a variable running through $\{\alpha_i\}$. Let further $\hat{v} = f(v)$ be a function such that: $\hat{\alpha}_i = \alpha_{i+1 (\mathrm{mod}\, k)}$ for any $\alpha_i \in \{\alpha_i\}$. Let now the set $\{\alpha_0, \alpha_1, \ldots, \alpha_{k-1}\}$ be replaced by the integers $0, 1, 2, \ldots, k-1$ put in a deliberate but fixed order. Let the function $\hat{v}$ be redefined accordingly. Let the particular type of set be symbolised by: $\{I_0, I_1, I_2, \ldots, I_{k-1}\}$. It is then obvious that given any such set $\{I_0, I_1, I_2, \ldots, I_{k-1}\}$, and replacing in all the definitions, Lemmata and Theorems dealt with so far, the set of values $\{0, 1, 2, \ldots, k-1\}$ by such a set, the respective results will hold equally for the set $\{I_i\}$ and respectively redefined functions.

Now, having a complete set based on a set of values $\{I_0, I_1, \ldots, I_{k-1}\}$, one can always generate a complete set of functions based on a different set of values, the reason being just the completeness of the first set. However, viewing the new set of functions from the point of view of the original set of functions, one would get a class of transformation.

This does not exhaust the possibilities: The basic types of functions: ⊞, ⊕, ⊙ may be defined not only with respect to 0, 1 (or I_0, I_1 for that matter), but also with respect to a general pair of values a, b (or I_a, I_b), whereas the a, b may be either the same with each function, or they may differ: In that case there will be specific rules applying to forming the formulae. These questions will be partly dealt with in another paper.

REFERENCES

[1] L. Kohout, "The Pinkava many-valued complete logic systems and their applications in the design of many-valued switching circuits", Proceedings of 1974 International Symposium on Multiple-Valued Logic, West Virginia.

[2] V. Pinkava, "A family of complete sets of functions in k-valued logics allowing easy generating", submitted to IEEETC.

[3] E. L. Post, "Introduction to a general theory of elementary propositions", Amer. J. Math., 43, pp. 163-185, 1921.

[4] D. A. Pospelov, Logičeskie metody analiza i sinteza sxem., Moscow, 1968.

[5] J. B. Rosser & A. R. Turquette, "Many-valued Logics, North-Holland, 1952.

[6] I. I. Žegalkin, "Aritmetizacija simvoličeskoi logiki", Matemat. Sbornik, 35, pp. 311-378, 1928 and 36, pp. 205-338, 1929.

TERNARY TWO-PLACE FUNCTIONS THAT ARE COMPLETE WITH CONSTANTS

J.C. Muzio
University of Manitoba
Winnipeg, Manitoba, Canada

1. Introduction and Notation

Let $E = \{0,1,2\}$ and denote by A the set of all functions with values in E and variables over E. A subset B of A is called complete if every member of A can be defined by a finite composition of members of B. If B is complete and only contains one element then that element is a Sheffer function. Let C be the set containing the three constant functions. If B is complete and $B = D \cup C$, $D \cap C = \Phi$ then D is called complete with constants, the same term being used if D consists of a single function. In this paper we are concerned entirely with single functions which are complete with constants. Some authors have used slightly different terminology for this concept, Wesselkamper [14] calling such functions weakly complete, while Rose [8] uses the term pseudo-Sheffer for those functions which are complete with constants but not Sheffer.

In this paper we are concerned entirely with ternary two-place functions and the conditions under which they are complete with constants. Much previous work has been on ternary Sheffer functions and necessary and sufficient conditions for such functions ([4], [5], [6], [13], [15] for example). It is well known that of the 19,683 two-place ternary functions, 3,774 are Sheffer. However comparatively little work has been done on functions which are complete with constants, although in a practical situation these are just as useful as Sheffer functions, since the constant functions will always be available. Two approaches to conditions for such functions will be given, the first to give a set of four necessary and sufficient conditions based on the brilliant general conditions of Rosenberg [9]. The second method uses an extension of the ideas of Kalicki [3] and the technique of [5] to give alternative necessary and sufficient conditions in terms of the one-place functions generated, and the classes in which they lie. For functions to be complete with constants it is sufficient to be able to generate two permutations which form a basis of S_3, the symmetric group on three marks, and a one-place function that takes exactly two values (Piccard's result [7], modified by Martin [4]).

Using the latter technique a systematic study of all two-place ternary functions is conducted. Clearly functions whose value set does not include 0, 1, and 2 cannot be complete with constants and these are called trivial. Further the functions of interest must be non-degenerate. These non-degenerate, non-trivial two-place ternary functions (18,138 of the 19,683) are divided into 1,587 equivalence classes of functions which are themselves split into seven types. For each of the seven types exact values for the numbers of functions which are complete with constants are given, the total number being 15,201.

Let p,q be variables over E and let Fpq be a two-place function. Define $H = C \cup \{F\}$, so that we are interested in the completeness of H. The type of F is defined to be $[\alpha\ \beta\ \gamma]$ where $\alpha = F00$, $\beta = F11$ and $\gamma = F22$. Then, allowing permutations of the values, the following seven types cover all the functions:

I	[0 0 0] ,
II	[1 0 0] ,
III	[0 0 1] ,
IV	[0 0 2] ,
V	[1 2 0] ,
VI	[0 2 1] ,
VII	[0 1 2] .

Hence, for example, [1 2 2] is of type III under the interchange of the values 0,2. The equivalence class which includes Fpq also includes Fqp and any functions which may be deduced from them by permutations of the value set. Consequently the maximum number of functions in an equivalence class is 12.

2. Rosenberg's Conditions

In 1966 Rosenberg gave a general set of six necessary and sufficient conditions for any set of *functors to be* complete (result in [9] and the proof in [10]). His conditions may be simplified in our case to give the following result. Initially his final condition does not apply in this case and his second condition concerning self-dual functions is immediately satisfied by the inclusion of the constant functions (as shown in Wesselkamper [14]). His remaining four conditions can be simplified a little for the restricted use we are making of them.

A relation $\lesssim$ is a partial order on E if it is reflexive, transitive and antisymmetric. F is monotonic with respect to $\lesssim$ if for all values $p_i \lesssim q_i$ $(i = 1,2;\ p_i,q_i \in E)$ then $Fp_1p_2 \lesssim Fq_1q_2$.

A subset of the Cartesian product $E \times E$ is a relation of degree 2 on E. Consider three such relations R_i $(i = 0,1,2)$ with the following properties $(p_i,q_i \in E;\ i = 1,2)$:

(i) if $\langle p_1,p_2\rangle \in R_i$ then $\langle p_2,p_1\rangle \in R_i$;

(ii) $\langle p_1,p_1\rangle \in R_i$;

(iii) if $\langle p_1,p_2\rangle,\langle q_1,q_2\rangle \in R_i$ then

$$\langle Fp_1q_1,Fp_2q_2\rangle \in R_i \quad .$$

$R_0 = \{\langle 0,1\rangle,\langle 0,2\rangle\}$; $R_1 = \{\langle 0,1\rangle,\langle 1,2\rangle\}$; $R_2 = \{\langle 0,2\rangle,\langle 1,2\rangle\}$.
Consequently from the definition R_0 contains all $\langle p_1,p_2\rangle$ except $\langle 1,2\rangle$, $\langle 2,1\rangle$.

We say F is linear on E if there exist n_1, n_2, n_3 such that for all $p_1, q_1 \in E$:

$$Fp_1q_1 = n_1p_1 + n_2q_1 + n_3 \quad \text{(modulo 3)} \ .$$

Consider a partition of the three values into two non-empty disjoint classes where $p_1 \sim p_2$ indicates both p_1, p_2 are in the same class $(p_1, p_2 \in E)$. If for one of the three possible such partitions F is such that whenever $p_1 \sim q_1$ and $p_2 \sim q_2$ then the values represented by Fp_1p_2 , Fq_1q_2 are in the same class then F has the proper substitution property.

The result may now be stated as:

Theorem 2.1. F is complete with constants if and only if the following four conditions hold:

1. For each partial order relation on E with a greatest and least element F is not monotonic with respect to this relation.
2. F is not linear on E .
3. F does not have the proper substitution property.
4. $\langle 1,2\rangle \in R_0$, $\langle 0,2\rangle \in R_1$, and $\langle 0,1\rangle \in R_2$.

Although we could have used this result to investigate all the ternary two-place functions it was found easier to deduce the results using the one-place functions as shown in the next section.

3. Class Investigation Technique

We shall use ideas based on the class definitions of Kalicki [3] which were used to investigate complete functors in [5]. These are defined as follows:

Let Gp be a well-formed formula (wff) in the single variable p and F be a two-place functor. Substitute the three values 0, 1, 2 in succession for p and suppose that Gp assumes the values p_0, p_1, p_2 then $p_0\ p_1\ p_2$ is called the value sequence of Gp , written $(p_0\ p_1\ p_2)$. In what follows it is convenient to identify the one-place functions by their value sequences and no ambiguity will arise from so doing.

The classes CL_i are defined by:

(i) CL_1 contains the variable p and the three constant functions.

(ii) CL_{s+1} contains all the wff's which can be built up by means of the generating functor F from constituent wff's of which one is an element of CL_s and the other is an element of CL_r $(r \leq s)$.

$|CL_s|$ is used to denote the set of value sequences of elements of CL_s. For convenience we define $DL_i = \bigcup_{r=1}^{i} CL_r$ and $|DL_i| = \bigcup_{r=1}^{i} |CL_r|$. Clearly if CL_s is empty then so is CL_r for all $r \geq s$.

Now a well known result of Słupecki [12] (also proved independently by Butler [1]) is that a set of functions which includes all one-place functions and a non-trivial, non-degenerate two-place function is complete. This would yield the immediate test that H is complete iff $|DL_{24}|$ contains all the one-place functions. However an immediate improvement is obtained from the result of Piccard [7] modified by Martin [4] that to generate all the one-place functions it is only necessary to generate two permuations that are a basis of the symmetric group S_3 and a one-place function which assumes exactly two values (see also Salomaa [11] and El Lozy [2] for related work). Salomaa [11] also proved that if we are concerned only with Sheffer functions rather than with ones which are complete with constants then it is sufficient to generate a basis of S_3. That this result does not carry over to functions which are complete with constants is shown by the following counter-example:

If Fpq is given by the table, we have

$|CL_1| = \{(012), (000), (111), (222)\}$,

$|CL_2| = \{(120), (201), (021)\}$,

$|CL_3| = \{(210), (102)\}$,

$|CL_4|$ is empty .

Fpq (p \ q)	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

Hence, although we have two generators of S_3, we do not have a function which is complete with constants. In terms of theorem 2.1 F is linear on E.

The pair of generators of S_3 which we require consists of one of: (120), (201) ; and one of: (021), (210), (102). We define K as the set consisting of such a basis of S_3 and any one-place function which assumes exactly two values.

The approach for any given two-place table is to generate the classes $|CL_i|$ until one of two stopping conditions is satisfied:

(i) for some k, $K \subseteq |DL_k|$ in which case the table represents a function which is complete with constants;

(ii) for some k, $|CL_k|$ is empty in which case the table represents a function which is not complete with constants.

Following a systematic investigation of two-place functions the following results were obtained for a two-place function Fpq, with $H = C \cup \{F\}$, and F being non-trivial and non-degenerate.

Theorem 3.1. F is complete with constants iff $K \subseteq |DL_6|$.

Theorem 3.2. If F is complete with constants then all 27 one-place functions are contained in $|DL_9|$.

Theorem 3.3. If F is not complete with constants then all the one-place functions generated are in $|DL_7|$.

Theorem 3.4. The maximum number of one-place functions which can be generated by H if it is not complete is 17.

Clearly it is not necessary to look at all 19,683 functions in detail but only at one representative from each equivalence class, taking functions as equivalent under an interchange of the variables, or some relabelling of the values. This means that the maximum number of functions in an equivalence class is 12, when the function is non-commutative and fully conjugated. In addition only those classes containing non-trivial non-degenerate functions need by considered. For each of the types I to VII there are 162, 342, 342, 342, 129, 197, and 73 equivalence classes respectively to be considered. Further for particular types the limits in theorems 1, 2 and 3 may be improved. For example, if F is of type IV then if F is complete with constants, $K \subseteq |DL_5|$ and all 27 one-place functions will be contained in $|DL_7|$ while if it is not complete with constants then all one-place functions which it will generate are contained in $|DL_6|$. However the results in the theorems are best possible in the sense that for each a function can be exhibited which requires the number of classes given.

4. Results

The main numerical results are given in the following table which details the numbers of functions of each type which are Sheffer, complete with constants but not Sheffer, trivial, non-trivial but degenerate, and the remainder. The functions in the second column are exactly those that Rose [8] would label pseudo-Sheffer. It is interesting to note that of the 18,138 non-trivial non-degenerate two-place functions, 15,201 or about 84% are complete with constants.

Type	Sheffer	Complete with constants (and not Sheffer)	Trivial	Non-trivial Degenerate	Remainder	Total
I		1,470	381	0	336	2,187
II	2,334	1,314	384	0	342	4,374
III		3,528	384	0	462	4,374
IV		2,970	384	0	1,020	4,374
V	1,440	12	0	4	2	1,458
VI		1,755	0	6	426	2,187
VII		378	0	2	349	729
Totals	3,774	11,427	1,533	12	2,937	19,683

Table of numbers of two-place ternary functions.

REFERENCES

[1] J. Butler, "On complete and independent sets of operations in finite algebras", Pacific J. Maths., vol. 10, pp.1167-1179, 1960.

[2] H. El Lozy, "On the generation of all the one variable functions in M-valued logic", Conf. Recd. 1973 Int. Symp. on Multiple-Valued Logic, pp.119-126.

[3] J. Kalicki, "A test for the existence of tautologies according to many-valued truth tables", J. Symb. Logic, vol. 15, pp.182-184, 1950.

[4] N.M. Martin, "The Sheffer functions of 3-valued logic", J. Symb. Logic, vol. 19, pp.45-51, 1954.

[5] J.C. Muzio, "A decision process for 3-valued Sheffer functions I", Zeitschr. f. math. Logik, vol.16, pp.271-280, 1970.

[6] J.C. Muzio, "A decision process for 3-valued Sheffer functions II", Zeitschr. f. math. Logik, vol. 17, pp.97-114, 1971.

[7] S. Piccard, "Sur les bases du groupe symétrique et les couples de substitutions qui engendrent un groupe régulier", Paris, Libraire Vuibert, 1946.

[8] A. Rose, Computer Logic, London, John Wiley, 1971.

[9] I. Rosenberg, "La structure des fonctions de plusieurs variables sur un ensemble fini", C.R. Acad. Sci. (Paris), vol. 260, pp.3817-3819, 1965.

[10] I. Rosenberg, "Über die funktionale Vollständigkeit in der mehrwertigen Logiken", Rozpravy Cs. Akademie Ved, Series Math. and Nat. Sci., vol. 80, pp.3-93, 1970.

[11] A. Salomaa, "On the composition of functions of several variables ranging over a finite set", Annales Universitatis Turkuensis (Ser. AI), vol. 41, 1960.

[12] J. Słupecki, "Kryterium pełności wielowartościowych systemów logiki zdań", C.R. Soc. Sci. Varsovie, vol. 32, Cl.III, pp.102-109, 1939.

[13] R.F. Wheeler, "Complete connectives for the 3-valued propositional calculus", Proc. London Math. Soc. (3), vol. 16, pp.167-191, 1966.

[14] T.C. Wesselkamper, "Some completeness results for abelian semi-groups and groups", Proc. 1974 Int. Symp. on Multiple-Valued Logic, pp.393-400.

[15] S.V. Yablonskii, "Functional constructions in k-valued logic", Trudy Mat. Inst. V.A. Steklov, vol. 51, pp.5-142, 1958 (in Russian).

FUNCTIONAL COMPLETENESS IN

HETEROGENEOUS MULTIPLE-VALUED LOGICS

I.G. Rosenberg
Université de Montréal
Montreal, Canada

Abstract

The composition of gates with more than one input and/or output alphabet into combinatorial circuits leads to the study of compositions and completeness problems in heterogeneous algebras. Using a type of composition (different from the two studied previously) in the case of finite pairwise disjoint alphabets we establish a Galois connection between certain closed classes of operations and certain classes of finitary relations. For the completeness problem several maximal classes are derived.

§1. Introduction

In this paper we consider the composition of gates into combinatorial circuits (with a single output and without feedbacks) in the case when there are more than one input and/or output alphabets. The behaviour of a gate with input alphabets $A_{i_0},\ldots,A_{i_{n-1}}$ and output alphabet A_{i_n} can be described by a heterogeneous operation (function, map) $A_{i_0}\times\ldots\times A_{i_{n-1}} \to A_{i_n}$. The theory of heterogeneous algebras [1], although less developped than the homogeneous universal algebra theory, has several applications in algebra (e.g. modules or monoids acting on a set) and automata theory. The composition of two gates B and C is obtained by attaching the output of C (working in A_i) to the ℓ-th input of C (working in A_j) provided that the alphabets match (i.e. $A_i \subseteq A_j$). In this way the composition may have no input in alphabet A_i. For this case Pöschel introduces a special fictitious (dummy) input in A_i. Blochina, Burosh and Kudrjavcev [2,3,8] (see also Pöschel [11]) even assume that fictitious inputs can be added and removed at will. These additional assumptions, minor as they seem to be, lead to very different results. In the case of homogeneous algebras [4] (corresponding to one alphabet combinatorial circuits) and in the case of Pöschel's composition [11] the relations on the alphabets play an important role, namely each polynomial class (a set of operations closed under compositions and containing the selectors) is uniquely determined by its invariant relations, which in their turn, can be described internally as sets of relations closed under very natural operations. For the case of a finite number of finite and pairwise disjoint alphabets we obtain the same result.

In the second part of the paper we study the completeness problem. Again in the case of a finite number of finite and pairwise disjoint alphabets, a set of operations is complete if and only if it is contained in no maximal class. Several maximal classes are given. If the sets A_i are not pairwise disjoint, the situation seems to be more complicated.

§2. Preliminaries

Let $A = \{A_i \mid i \in I\}$ be a class of pairwise distinct sets, each called a *phylum* of A

and each having at least two elements. Given $i_0,\ldots,i_{n-1} \in I$ (n finite) call each mapping $f:A_{i_0}\times\ldots\times A_{i_{n-1}} \to \bigcup_{i\in I} A_i$ with $\underline{r}f := f(A_{i_0}\times\ldots\times A_{i_{n-1}}) \subseteq A_{i_n}$ for some $i_n \in I$, an (n-ary) *heterogeneous operation* of type $\underline{t}f := \langle i_0,\ldots,i_{n-1}\rangle$. Thus f assigns to each $x = \langle x_0,\ldots,x_{n-1}\rangle \in A_{i_0}\times\ldots\times A_{i_{n-1}}$ some value $fx = fx_0\ldots x_{n-1}$ in A_{i_n}. Note that the target set A_{i_n} is not part of the definition, i.e. it may happen that $\underline{r}f$ is a subset of several sets A_i. The class of all finitary heterogeneous operations is denoted by $\underset{\sim}{O}_A$ or shortly by $\underset{\sim}{O}$. Given $J \subseteq I$ and $j \in I$ denote by $\underset{\sim}{O}_{Jj}$ the class of all operations f of type $\underline{t}f = \langle i_0,\ldots,i_{n-1}\rangle$ with $i_0,\ldots,i_{n-1} \in J$ and $\underline{r}f \subseteq A_j$. If $J = \{i\}$ write $\underset{\sim}{O}_{ij}$ instead of O_{Jj}. Thus $\underset{\sim}{O}_{ii}$ is the set of all finitary *homogeneous operations* on A_i. A *heterogeneous algebra* is a pair $\langle A,F\rangle$ where $F \subseteq \underset{\sim}{O}_A$. There are several naturally defined heterogeneous algebras, e.g. monoids acting on a set, modules, automata (sequential machines) and state machines (semiautomata) [1]. As observed in [11], heterogeneous operations can describe the behaviour of gates (devices) with more than one input and output alphabet. To combine such gates into a combinatorial circuit (single output and without feedbacks) naturally we can connect an output of a gate only to an input of another gate whose alphabet contains the alphabet of the first gate. This leads to the following notion of a composition of heterogeneous operations. Let $f,g \in \underset{\sim}{O}_A$ be of the type $\langle i_0,\ldots,i_{n-1}\rangle$ and $\langle j_0,\ldots,j_{m-1}\rangle$, respectively, and let $\underline{r}g \subseteq A_{i_0}$ (in the sequel f and g always denote such operations). Then the *composition* $f*g$ of f and g is the heterogeneous operation of the type $\langle j_0,\ldots,j_{m-1},i_1,\ldots,i_{n-1}\rangle$ assigning the value

$$hx = f(gx_0\ldots x_{n-1})x_n\ldots x_{n+m-2}$$

to every $x \in A_{j_0}\times\ldots\times A_{j_{m-1}}\times A_{i_1}\times\ldots\times A_{i_{n-1}}$. Note that $\langle \underset{\sim}{O}_A;* \rangle$ is a partial semigroup. We extend * to a complete binary operation on $\underset{\sim}{O}_A$ by setting $f*g = f$ whenever $\underline{r}g \not\subseteq A_{i_0}$ (however then * is no longer associative). It should be noted that the composition (called superposition) introduced by R. Pöschel [11] differs from ours. Namely his superposition (without information loss) contains a fictitious (dummy or non-essential) variable x_{i_0}, i.e. h is of type $\langle j_0,\ldots,j_{m-1},i_0,\ldots,i_{n-1}\rangle$ and to each $x \in A_{j_0}\times\ldots\times A_{j_{m-1}}\times A_{i_0}\times\ldots\times A_{i_{n-1}}$ assigns $hx = fgx_0\ldots x_{m-1}x_{m+1}\ldots x_{m+n-1}$. In this way h carries the information that the first input of f is of the type i_0.

For each finite sequence $\iota := \langle i_0,\ldots,i_{n-1}\rangle$ of elements from I and $0 \le k < n$ define the k-th *selection* $e^\iota_k:A_{i_0}\times\ldots\times A_{i_{n-1}} \to A_{i_k}$ *of type* ι by setting $e^\iota_k x = x_k$ for every $x \in A_{i_0}\times\ldots\times A_{i_{n-1}}$.

Naturally we should allow the renumbering (permutation) of variables. Following the idea from [10] we define two unary operations ζ and τ on $\underset{\sim}{O}_A$ iterations of which produce all permutations of variables. For a unary operation f set $\zeta f = \tau f = f$. For $n > 1$ define (i) ζf (of type $\langle i_{n-1},i_0,\ldots,i_{n-2}\rangle$) by setting $(\zeta f)x = fx_1\ldots x_{n-1}x_0$ for every $x \in A_{i_{n-1}}\times A_{i_0}\times\ldots\times A_{i_{n-2}}$, and (ii) τf (of type $\langle i_1,i_0,i_2,\ldots,i_{n-1}\rangle$) by setting $(\tau f)x = fx_1x_0x_2\ldots x_{n-1}$ for every $x \in A_{i_1}\times A_{i_0}\times A_{i_2}\times\ldots\times A_{i_{n-1}}$. Since the cycle $(0,\ldots,n-1)$ and transposition $(0,1)$ generate all permutations of $\{0,\ldots,n-1\}$, by repeated application of ζ and τ we can obtain from f each operation differing from it by the order of variables only.

Naturally we should also allow the identification of variables of the same type (i.e. ranging over the same phylum). To achieve this we define the following unary

operation Δ on $\underset{\sim}{O}_A$. If $i_0 \neq i_1$ or $n = 1$ set $\Delta f = f$. If $n > 1$ and $i_0 = i_1$ define Δf (of type $\langle i_0, i_2, \ldots, i_{n-1} \rangle$ by setting $(\Delta f)x = fx_0x_0x_1, \ldots, x_{n-2}$ for every $x \in A_{i_0} \times A_{i_2} \times \ldots \times A_{i_{n-1}}$. By (repeated) application of ζ, τ and Δ we can get every operation obtainable from f by rearrangement of variables and by identification of variables (of the same type).

Each subclass of $\underset{\sim}{O}_A$ closed with respect to $*$, ζ, τ and Δ is called a *closed* (or *preiterative*) *class* (i.e. it is a subalgebra of $\langle \underset{\sim}{O}_A; *, \zeta, \tau, \Delta \rangle$ provided $\underset{\sim}{O}_A$ is a set). We can introduce the following unary operation ∇_i ($i \in I$) which adds x_i as a fictitious variable. Given f of type $\langle i_0, \ldots, i_{n-1} \rangle$ define $\nabla_i f$ of type $\langle i_0, \ldots, i_{n-1}, i \rangle$ by setting $(\nabla_i f)x = fx_0 \ldots x_{n-1}$ for every $x \in A_{i_0} \times \ldots \times A_{i_{n-1}} \times A_i$. A closed class which is also closed with respect to all operations ∇_i is called *iterative*. A closed class containing all selectors is called a *polynomial class*. Clearly each polynomial class is iterative. Finally we introduce the following operation ∂ for the removal of fictitious variables (the i-th variable is fictitious if $fx' = fx''$ whenever $x'_j = x''_j$ for all $0 \leq j < n$, $j \neq i$). If the first variable of f is not fictitious or $n = 1$ set $\partial f = f$. If the first variable is fictitious define ∂f (of type $\langle i_1, \ldots, i_{n-1} \rangle$) by setting $(\partial f)x = fax_0 \ldots x_{n-2}$ for every $x \in A_{i_1} \times \ldots \times A_{i_{n-1}}$ (where a is any element of A_{i_0}). Every iterative class closed with respect to ∂ is called *strongly iterative*. Thus in a strongly iterative class we can add and remove fictitious variables at will, e.g. if it contains $h: A_{i_0} \to \{a\}$ then it contains each $h': A_i \to \{a\}$. Strongly iterative classes are called $*$-Post algebras in [11]. If A is a set, then the closed, iterative, polynomial, and strongly iterative classes are the subalgebras of suitable defined algebra on $\underset{\sim}{O}_A$; hence they form an algebraic lattice which we denote by $\mathcal{L}_A$, $\mathcal{L}^i_A$, $\mathcal{L}^p_A$, and $\mathcal{L}^{si}_A$, respectively. For $|I| = 2$ and $|A_1| = |A_2| = 2$ it is shown in [8,3] that $\mathcal{L}^{si}_A$ is more complicated than Post's lattice of closed classes of Boolean functions. However it is dually atomic (the papers contain the full lists of all dual atoms). Of course, the dual atoms of $\mathcal{L}^{si}_A$ need not to be dual atoms of $\mathcal{L}_A$.

§3. Relations

In the case of homogeneous algebras ($I = \{i\}$) the polynomial classes can be described in terms of relations on the set A_i [4,16,17,18]. For heterogeneous algebras with finite phyla R. Pöschel [11] characterized closed classes (with respect to his type of composition) in terms of relations. A similar approach is used in our case. Let J be a non-void set and let $\underset{\sim}{R}^{(J)}$ be the class of all maps $J \to A_i$ ($i \in I$). Each subclass ρ of $\underset{\sim}{R}^J$ is called a J-relation on A. For every cardinal $\underline{a}$ let $\underline{a}$ be the set of all ordinals of cardinality less than $\underline{a}$ (e.g. $\underline{2} = \{0,1\}$). In most of the cases we use $J = \underline{a}$ and refer to the $\underline{a}$-relations on A as the $\underline{a}$-ary relations. For a finite n we identify the sets $B^{\underline{n}}$ with the cartesian power B^n and the $\underline{n}$-relations on B with the usual n-ary relations on B. For each $i \in I$ let ρ_i stand for $\rho \cap A_i^J$. We say that f *preserves* ρ (or is *invariant* or *stable* w.r.t. ρ) if $r = fr_0 \ldots r_{n-1} \in \rho$ whenever for all $j = 0, \ldots, n-1$ we have $r_j \in \rho_j$ (here and in the sequel r is the map $J \to A_{i_n}$ defined by $rj = f(r_0 j) \ldots (r_{n-1} j)$ for every $j \in J$).

3.1. Examples. Let $I = \underline{2}$, $A_0 = \{0,1\}$ and $A_1 = \{0,2\}$ [3].
1) Consider the unary relation (subset of $A_0 \cup A_1$) $\rho = \{0,2\}$. Then f of type $\langle 0,1 \rangle$ preserves ρ iff $f(\{0\} \times A_1) \subseteq A_1$.

2) Consider the binary relation $\sigma = \{\langle 0,1\rangle, \langle 2,0\rangle\}$. Then f of type $\langle 0,1\rangle$ preserves σ iff $\langle f02, f10\rangle \in \sigma$.

3.2. Remark. If $\rho_{i_j} = \emptyset$ for some $0 \leq j < n$, then trivially f preserves ρ. On the other hand, if all $\rho_{i_j} \neq \emptyset$ while $\rho_{i_n} = \emptyset$, then trivially f does not preserve ρ; in this respect we differ from [11] where in this case f preserves ρ.

As in the case of homogeneous algebras ([4,17,18]) the correspondence "f preserves ρ" induces a Galois connection between polynomial classes and certain sets of relations on $\underset{\sim}{A}$. We restrict our attention to the case of $\underset{\sim}{A}$ with all phyla finite. Let $\underset{\sim}{R}_A^{(n)}$ (or shortly $\underset{\sim}{R}^{(n)}$) be the class of all n-ary relations on $\underset{\sim}{A}$ $(n = 1,2,\ldots)$ and let $\underset{\sim}{R}_A = \bigcup_{n=1}^{\infty} \underset{\sim}{R}_A^{(n)}$. For any subclass Y of $\underset{\sim}{R}_A$ let Pol Y be the class of all $f \in \underset{\sim}{O}_A$ preserving every $\rho \in Y$. Conversely for each subclass $X \subseteq \underset{\sim}{O}_A$ let Inv X be the class of all relations $\rho \in \underset{\sim}{R}_A$ preserved by every $f \in X$. Let $\underset{=}{P}_A$ and $\underset{=}{I}_A$ be the families of all classes of the type Pol Y $(Y \subseteq \underset{\sim}{R}_A)$ and Inv X $(X \subseteq \underset{\sim}{O}_A)$, respectively. It follows from the general theory of Galois connections that $\phi: X \to \text{Inv } X$ and $\psi: Y \to \text{Pol } Y$ are (mutually inverse) anti-isomorphisms of $\underset{=}{P}_A$ onto $\underset{=}{I}_A$ and $\underset{=}{I}_A$ onto $\underset{=}{P}_A$, respectively. Moreover, $\text{Pol}(Y_1 \cup Y_2) = \text{Pol } Y_1 \cap \text{Pol } Y_2$, $Y \subseteq \text{Inv Pol } Y$, $\text{Inv}(X_1 \cup X_2) = \text{Inv } X_1 \cap \text{Inv } X_2$, and $X \subseteq \text{Pol Inv } X$.

Now we characterize $\underset{=}{P}_A$ as the family of all polynomial classes. A direct check shows that each $X \in \underset{=}{P}_A$ is a polynomial class. In order to prove that each polynomial class X belongs to $\underset{=}{P}_A$ we introduce the notion of the t-orbit of X (which generalizes the notion of the n-th graph from [4,11]. Let $t = \{t_0,\ldots,t_{n-1}\} \subseteq \underset{\sim}{R}^{(h)}$ be such that $t_\ell \in A_{i_\ell}^h$ $(\ell = 0,\ldots,n-1)$. Let X be a polynomial class. The relation

$$\Omega_X^t = \{ft_0\ldots t_{n-1} \mid f \in X, \underline{t}f = \langle i_0,\ldots,i_{n-1}\rangle\}$$

is called the t-*orbit* of X. Note that the order of the elements of t is immaterial because the polynomial class X is closed with respect to the permutation of variables. We say that t is *surjective* (*of type* $\iota = \langle i_0,\ldots,i_{p-1}\rangle$) if

$$A_{i_0} \times \ldots \times A_{i_{n-1}} = \{\langle t_0\ell,\ldots,t_{p-1}\ell\rangle \mid \ell \in \underline{h}\}.$$

We say that a system $\{\sigma_k \mid k \in K\}$ of relations from $\underset{\sim}{R}^{(h)}$ is *updirected* if to each $\{k_0,k_1\} \subseteq K$ there exists $k_2 \in K$ such that $\sigma_{k_i} \subseteq \sigma_{k_2}$ $(i = 0,1)$. We have

3.3. Proposition. *Let* $t = \{t_0,\ldots,t_{n-1}\}$ *where* $t_\ell \in A_{i_\ell}^h$ $(\ell = 0,\ldots,n-1)$ *and let* X *be a polynomial class. Then:*

(i) *If either* t *is surjective or all* A_i *are pairwise disjoint, then* Ω_X^t *is the least relation from* $\underset{\sim}{R}^{(h)} \cap \text{Inv } X$ *containing* t.

(ii) *If all* A_i *are pairwise disjoint, then each* $\rho \in \underset{\sim}{R}^{(h)} \cap \text{Inv } X$ *is the union of an (updirected) system of relations* Ω_X^t.

(iii) *If* t *is surjective and* $f \in \text{Pol } \Omega_X^t$ *is of type* $\langle i_0,\ldots,i_{n-1}\rangle$, *then* $f \in X$.

3.4. Proposition. *Let all* A_i *be finite and let* $X \subseteq \underset{\sim}{O}_A$. *Then the following conditions are equivalent*

(i) X = Pol Inv X,

(ii) X = Pol Y *for some* $Y \subseteq \underset{\sim}{R}_A$,

(iii) X *polynomial class*.

For every set B, $h > 0$ and $b = \langle b_0,\ldots,b_{h-1}\rangle \in B^h$ set $\ker b = \{\langle i,j\rangle \in \underline{h} | b_i = b_j\}$. Thus ker b is an equivalence relation on $\underline{h}$. Given an arbitrary equivalence relation ε on $\underline{h}$ set

$$\Delta_{b\varepsilon} = \{b \in B^h | \ker b \supseteq \varepsilon\}$$

and $\Delta_{A\varepsilon} = \bigcup_{i\in I} \Delta_{A_i\varepsilon}$. The relations $\Delta_{A\varepsilon}$ and the relation $\emptyset$ are called *diagonal*. As in the homogeneous case [15,18] we have:

3.5. Proposition. *Let* $\rho \in \underset{\sim}{R}$. *Then* Pol $\{\rho\} = \underset{\sim}{O}_A$ *if and only if* ρ *is diagonal.*

§4. Operations on relations

In the homogeneous case the sets Inv X are internally characterized as the subalgebras of a naturally defined algebra (called *Post coalgebra* in [4] and *composition coalgebra* [17,18]) on the set $\underset{\sim}{R}$ of all relations. For heterogeneous algebras R Pöschel [11] obtained a similar result (for his type of composition). In our situation we can adopt all the operations except one which needs to be modified. We have selected the following operations on $\underset{\sim}{R}$:

1) The zero (nullary) operations $\emptyset$ and A (i.e. having the values the relation $\emptyset$ and the unary relation $A = \bigcup_{i\in I} A_i$, respectively).

2) The unary operations ζ, τ, ∂ and Δ which to every $\rho \in \underset{\sim}{R}^{(h)}$ $(h > 1)$ assign

$$\zeta\rho = \{\langle a_1,\ldots,a_{h-1},a_0\rangle | a \in \rho\},$$
$$\tau\rho = \{\langle a_1,a_0,a_2,\ldots,a_{h-1}\rangle | a \in \rho\},$$
$$\partial\rho = \{\langle a_0,a_2,\ldots,a_{h-1}\rangle | a \in \rho, a_0 = a_1\},$$
$$\Delta\rho = \{a \in \rho | a_0 = a_1\};$$

and such that $\zeta\rho = \tau\rho = \delta\rho = \Delta\rho$ for $\rho \in \underset{\sim}{R}^{(1)}$.

3) The binary relation $\times$ (*cartesian product*) which to every $\rho' \in \underset{\sim}{R}^{(h')}$ and $\rho'' \in \underset{\sim}{R}^{(h'')}$ assigns

$$\rho' \times \rho'' = \{\langle a'_0,\ldots,a'_{h'-1},a''_0,\ldots,a''_{h''-1}\rangle | a' \in \rho', a'' \in \rho''\}.$$

4) The operation $\cap$ which to every non-void system $\rho = \langle \rho^k | k \in K\rangle$ of h-ary relations on A assigns the h-ary relation $\sigma = \cap\rho$ with $\sigma_i = \bigcap_{k\in K} \rho^k_i$ for every $i \in I$.

5) The operation $\cup\hat{}$: let $\rho = \langle \rho^k | k \in K\rangle$ be a non-empty system of h-ary relations on A. If ρ is updirected, set $\cup\hat{}\rho = \cup\langle \rho^k | k \in K\rangle$. If ρ is not updirected, set $\cup\hat{}\rho = A^h$.

A straight verification shows that each class Inv X contains $\emptyset$ and A. Further for $\rho,\sigma \in \underset{\sim}{R}$ we have Pol ρ = Pol $\zeta\rho$ = Pol $\tau\rho$, Pol $\rho \subseteq$ Pol $\Delta\rho$, Pol $\rho \subseteq$ Pol $\partial\rho$ and Pol $\rho \cap$ Pol $\sigma \subseteq$ Pol $(\rho\times\sigma)$ (the last with = provided all ρ_i and σ_i are non-void). Moreover for every non-empty system $\rho = \langle \rho^k | k \in K\rangle$ of h-ary relations we have $\bigcap_{k\in K}$ Pol $\rho^k \subseteq$ Pol $\cap\rho$ and $\bigcap_{k\in K}$ Pol $\rho^k \subseteq$ Pol $\cup\hat{}\rho$. Thus for each $X \subseteq \underset{\sim}{O}_A$ the class Inv X is closed w.r.t. the operations 1)-5).

In the homogeneous case we have also the unary operation r (restriction) which deletes the last coordinate: $r\rho = \{ \langle a_0,\ldots,a_{h-2} \rangle \mid a \in \rho\}$ for every $\rho \in \underset{\sim}{R}^{(h)}$ with $h > 1$ and $r\rho = \rho$ for $\rho \in \underset{\sim}{R}^{(1)}$. In the heterogeneous case, if the sets A_i are not pairwise disjoint, there are sets Inv X which are not closed with respect to ρ. To see this, consider the relations ρ and σ from 3.1. Clearly $\rho = r\sigma$. We show that $\rho \notin$ Inv Pol σ. Define $f \in \underset{\sim}{O}_A$ of type $\langle 0,1 \rangle$ by setting $f02 = 0$ and $fx = 1$ otherwise. Clearly $\langle f02, f10 \rangle = \langle 0,1 \rangle \in \sigma$, hence $f \in$ Pol σ while $f \notin$ Pol ρ (because $f00 = 1 \notin \rho$). Thus we must modify the definition of r.

6) The unary operation r': Let $\rho \in \underset{\sim}{R}^{(h)}$ $(h > 1)$ and let $\sigma = \{ \langle a_0,\ldots,a_{h-2} \rangle \mid a \in \rho\}$. If

$$\sigma_i \cap A_j^h \subseteq \sigma_j \text{ for every } \{i,j\} \subseteq I \tag{1}$$

set $r'\rho = \sigma$. If (1) does not hold or $\rho \in \underset{\sim}{R}^{(1)}$ set $r'\rho = \rho$.

Note that (1) trivially holds if all A_i are pairwise disjoint. We have:

4.1. Proposition. *Each class* Inv X *is closed with respect to* r'.

The structure $\mathcal{R} = \langle \underset{\sim}{R};\emptyset,A,\zeta,\tau,\partial,\Delta,\cap,\hat{\cup},r',\times \rangle$ is called the *full composition coalgebra*. Each subclass of $\underset{\sim}{R}$ closed with respect to the operations of $\mathcal{R}$ is called a *composition coalgebra*.

It is easy to prove:

4.2. Proposition. *Let* $X \subseteq \underset{\sim}{O}_A$. *Then* Inv X *is a composition coalgebra.*

Following the method of proof [7] (used also in [11]) one can prove:

4.3. Proposition. *Let* C *be a composition coalgebra and let* t *be a finite surjective subset of* $\underset{\sim}{R}^{(h)}$. *Then* $\Omega^t_{\text{Pol C}}$ *belongs to* C.

Now for finite pairwise disjoint A_i's we obtain immediately:

4.4. Proposition. *Let all* A_i *be finite and pairwise disjoint. Then for each* $Y \subseteq R$ *the following conditions are equivalent:*

(i) Y = Inv Pol Y,

(ii) Y = Inv X *for some* $X \subseteq \underset{\sim}{O}_A$,

(iii) Y *is a composition coalgebra.*

§5. Maximal classes

For every $X \subseteq \underset{\sim}{O}_A$ let [X] be the least closed class containing X. We say that X is *complete* if $[X] = \underset{\sim}{O}_A$. Now we give a simple completeness criterion for sets X such that $[X] \supseteq \underset{\sim}{O}_{ij}$ (the set of all $g:A_i^m \to A_j, m = 1,2,\ldots$). For every set B we set $\omega(B) = \{ \langle b,b \rangle \mid b \in B\}$ (least equivalence on B). We assume that all A_i are finite.

5.1. Proposition. *Let* $X \subseteq \underset{\sim}{O}_A$ *and let* $[X] \supseteq \underset{\sim}{O}_{ij}$ *for some* $i,j \in I$. *Then* X *is complete if and only if*

(i) *For every* $k \in I\backslash\{i\}$

$$\omega(A_k) = \cap\{\ker \phi \mid \phi \in [X], \phi: A_k \to A_i\},$$

and

(ii) *For every* $\ell \in I\backslash\{j\}$ *there is* $g \in \underset{\sim}{O}_{j\ell} \cap [X]$ *with* $\underline{\underline{r}}g = A_\ell$.

We say that a closed class M is *maximal (precomplete) in* $\underset{\sim}{O}_A$ if $M \subset \underset{\sim}{O}_A$ and $M \subset C \subset \underset{\sim}{O}_A$ for no closed class C. Thus if A is a set, M is a dual atom or coatom in the lattice $\mathcal{L}_A$ of all closed classes. Naturally we might ask whether $\mathcal{L}_A$ is dually atomic (i.e. each proper class is contained in a maximal class). If it is, then the list of all maximal classes provides the most general completeness criterion: $X \subseteq \underset{\sim}{O}_A$ is complete if and only if it is contained in *no* maximal class.

It is easy to establish:

5.2. Proposition. *To each closed class* $C \subset \underset{\sim}{O}_A$ *there exists a polynomial class* M *such that* $C \subseteq M \subset \underset{\sim}{O}_A$. *Each maximal class is polynomial.*

For unary relations ρ on A (i.e. subsets of A) we can determine all maximal classes of the form Pol ρ.

5.3. Proposition. *Let* $\rho \subset A$ *be non-void. Then* Pol ρ *is maximal if and only if either*

(i) *all* $\rho_i \neq \emptyset$, *or*

(ii) $\rho = \cup_{j\in J} A_j$, *where* $\emptyset \subset J \subset I$ *and* $\rho \cap A_i = \emptyset$ *for all* $i \in I\backslash J$, *or*

(iii) *there exist* $\rho_i = \emptyset$, $\rho_j \neq \emptyset$ *and* $A_i \cap A_j \neq \emptyset$.

In the homogeneous case the semigroup $X^{(1)}$ (under composition of maps) of all unary operations from a closed class X (called sometimes the "foundation" of X) is of some importance (e.g. Post [13] used it for the classification of all closed classes of Boolean functions). Consider the heterogeneous case. Suppose $I = \{0,\ldots,q-1\}$ and that all A_i are finite. Let X be a closed class. The set

$$\text{tr } X = \{h \in X \mid \underline{\underline{t}}h = \langle 0,\ldots,q-1 \rangle\}$$

will be called the *trace* of X. Let $t = \{t_0,\ldots,t_{q-1}\}$ be a surjective set of type $\langle 0,\ldots,q-1 \rangle$ (i.e. $t \subseteq \underset{\sim}{R}^{(h)}$ where $h = |A_0|\ldots|A_{q-1}|$). The relation Ω^t_X contains all information on tr X. By 3.3(i) we know that for a polynomial class X we have $X \subseteq \text{Pol } \Omega^t_X$. Thus each maximal class M is either of the form Pol Ω^t_X or satisfies Pol $\Omega^t_M = \underset{\sim}{O}_A$. Consider the latter case. Then Ω^t_M is diagonal. From $\Omega^t_M \supseteq t$ it follows (by 3.5) that $\Omega^t_M = \cup_{i=0}^{q-1} A_i^h$ which implies that tr M is the set Q of all $h \in \underset{\sim}{O}_A$ of type $\langle 0,\ldots,q-1 \rangle$ (i.e. tr $\underset{\sim}{O}_A$). This leads naturally to the question: What are the maximal polynomial classes containing Q? The well-known *Slupecki criterion* [20] (cf. also [5]) answers a similar question for the finite homogeneous algebras containing all unary operations (a unique class). For countable homogeneous algebras Gavrilov [6] showed that there are precisely two maximal classes containing all unary operations. In the case that all A_i are pairwise disjoint we have:

5.4. Proposition. *Let* $I = \{0,\ldots,q-1\}$ *and assume that the* A_i*'s are pairwise disjoint. Let* Q *be the set of all operations of the type* $\langle 0,\ldots,q-1 \rangle$. *Then the lattice* $\mathcal{L}^t$ *of all polynomial classes containing* Q *is dually atomic. Each dual atom of* $\mathcal{L}^t$ *has the*

form Pol ρ *where* $\rho \neq \emptyset$ *is unary such that at least one* $\rho_i = \emptyset$.

5.5. Corollary. *If* I *is finite and all* A_i *are pairwise disjoint, then the lattice of all closed classes is dually atomic and has a finite number of atoms.*

Now we proceed to list a series of maximal classes. We assume throughout that all A_i are finite. The first result generalizes the classes $O(A_i)$ from [8,3].

5.6. Proposition. *Let* $\emptyset \subset J \subset I$. *Suppose* $\rho_j = A_j^2$ *for every* $j \in J$ *and* $\rho_i = \omega(A_i)$ *for every* $i \in I \setminus J$. *Then* Pol ρ *is maximal.*

The equivalence relations $\omega(B) = \{\langle b,b\rangle \mid b \in B\}$ and B^2 on a set B are said to be *trivial*. We have

5.7. Proposition. *Let* $\emptyset \subset J \subset I$. *For every* $j \in J$ *let* ρ_j *be a nontrivial equivalence relation on* A_j. *For every* $i \in I \setminus J$ *let* $\rho_i = \omega(A_i)$. *Then* Pol ρ *is a maximal class.*

The following maximal classes generalize the class S from [8,3]:

5.8. Proposition. *Let* $\emptyset \subset J \subset I$ *and let* p *be a prime divisor of all* $|A_j|$ $(j \in J)$. *For every* $j \in J$ *let* $\rho_j = \{\langle x, s_j x\rangle \mid x \in A_j\}$ *where* s_j *is a permutation of* A_j *with* $|A_j|/p$ *disjoint cycles of length* p. *For every* $i \in I \setminus J$ *let* $\rho_i = \omega(A_i)$. *Then* Pol ρ *is maximal.*

In the following proposition we determine a series of maximal classes with the property that each ρ_i is either A_i^h or the set $\mathrm{Pol}_i\, \rho_i$ (of all homogeneous $f \in \underset{\sim}{O}_{ii}$ preserving ρ_i) is maximal in $\underset{\sim}{O}_{ii}$. Although the formulation of the proposition is rather complicated, by specializing ρ_i we can obtain from F several maximal classes.

Let B be a finite set. By P_B we denote the set of all partial orders on B with a least and greatest element. By S_B we denote the set of all $\{\langle b,sb\rangle \mid b \in B\}$ where s is a permutation of B with $|B|/p$ disjoint cycles of the same prime length p. By L_B we denote the set of all relations $\{x \in B^4 \mid x_0 \dot{+} x_1 = x_2 \dot{+} x_3\}$ where $\dot{+}$ is a p-elementary (abelian) group operation (p prime, $\dot{+}$ commutative and pb is the zero element for every $b \in B$; it exists iff $|B| = p^k$). Let $M_B = P_B \cup S_B \cup L_B$. For each $\rho \in M_B$ the class Pol ρ is maximal in $\underset{\sim}{O}_{\{B\}}$ [9,21,14,15]. Let $\iota_B = \{x \in B^{|B|} \mid x_p = x_q$ for some $0 \leq p < q < |B|\}$. The class Pol ι_B (called Słupecki class) is maximal in $\underset{\sim}{O}_{\{B\}}$ [9,14].

5.9. Proposition. *Let* $h > 1$, $\rho \in \underset{\sim}{R}^{(h)}$, *and* $\emptyset \subset J \subseteq I$. *Suppose that for all* $\{j,j'\} \subseteq J$ *and* $i \in I \setminus J$ *we have*

(i) $J = I$ *and* $\omega(A_j) = \cap\{\ker \phi \mid \phi \in \mathrm{Pol}\ \rho, \phi: A_j \to A_{j'}\}$ *provided* $\rho_k \in M_{A_k}$ *for some* $k \in I$,

(ii) $\rho_j \neq \iota_{A_j}$ *and* $\mathrm{Pol}_j\, \rho_j$ *is maximal in* $\underset{\sim}{O}_{jj}$,

(iii) $\rho_i = A_i^h$,

(iv) *for every* $r \in \rho_j$, *there exist* $n > 0$, *elements* $r_0, \ldots, r_{n-1} \in \rho_j$, *and* $h \in \underset{\sim}{O}_{jj}^{\cap}$ Pol ρ *such that* $r = hr_0 \ldots r_{n-1}$,

(v) *there exist* $g \in \underset{\sim}{O}_{jj}^{\cap}$ Pol ρ *with* $\underline{\underline{r}}f = A_j$.

Then Pol ρ *is maximal.*

By specializing ρ_j ($j \in J$) to the six types of relations such that $\mathrm{Pol}_j \rho_j$ is maximal in $\underset{\sim}{O}_{jj}$ [14,15] it is possible to obtain several maximal classes which include many of the maximal classes from [8,3]. We will not attempt to do so here. It can be proved that in the case of pairwise disjoint A_i's each maximal class Pol ρ satisfies $\mathrm{Pol}_i \rho_i \subset \underset{\sim}{O}_{ii} \Rightarrow \mathrm{Pol}_i \rho_i$ maximal. This is not true if the sets A_i are not pairwise disjoint. The following example generalizes the class W from [8,3].

5.10. Proposition. *Let* $\rho_i = C_i \times D_i$ *where* C_i *and* D_i *are nonempty disjoint subsets of* A_i ($i \in I$). *Suppose that there exist* $i',i'',j',j'' \in I$ *such that*

$$C_{i'} \cap A_{j'} \not\subseteq C_{j'}, \quad D_{i''} \cap A_{j''} \not\subseteq D_{j''}.$$

Then Pol ρ *is maximal.*

Bibliography

[1] G. Birkhoff, J.D. Lipson, Heterogeneous Algebras, *J. Combinatorial Theory* 8 (1970) 115-133.

[2] G.N. Blochina, W.B. Kudrjavcev, G. Burosh, Über gewisse Eigenschaften des Systems P, *Math. Nachrichten* 54 (1972) 355-378.

[3] G.N. Blochina, V.B. Kudrjavcev, G. Burosch, Das Problem der Vollständigkeit für Boolesche Funktionen über zwei Dualmengen mit nichtleerem Durchsnitt I-II, *Zeitschr. f. math. Logik und Grundlagen Math.* 19 (1973) 163-180, 20 (1974) 79-96.

[4] V.G. Bodnarčuk, L.A. Kalužnin, V.N. Kotov, B.A. Romov, Galois Theory for Post Algebras (Russian), *Kibernetika* (Kiev) Part I: 3 (1969) 1-10, Part II: 5 (1969) 1-9. Engl. Translation *Cybernetics* 5 (1969) 243-252, 531-539.

[5] G.A. Burle, The classes of k-valued logic containing all functions of one variable (Russian), *Diskretnič Analiz* 10 (1967) 3-7.

[6] G.P. Gavrilov, On the functional completeness in countably-valued logics (Russian), *Problemy kibernetiky* 15 (1965) 5-64.

[7] L.A. Kalužnin, M.H. Klin, Certain maximal subgroups of symmetric and alternating groups (Russian), *Mat. Sbornik* 87 (129) (1972) 91-121, English translation *Math.* USSR-Sb 16 (1972) 95-124.

[8] W.B. Kurdjavcev, G. Burosh, Das Problem der Vollständigkeit für Boolesche Funktionen über zwei Dualmengen, *Math. Nachrichten* 54 (1972) 105-125.

[9] S.V. Jalbonskič, Functional constructions in the k-valued logic (Russian), *Trudy Mat. Inst. im. V.A. Steklova*, T.51 (1958) 5-142.

[10] A.I. Mal'cev, Iterative algebras and Post's varieties (Russian), *Algebra i Logika* 5 (1966) 5-24.

[11] R. Pöschel, Postsche Algebren von Funktionen über einer Familie endlicher Mengen, *Zeitschr. f. Math. Logik und Grundlagen d. Math.* 19 (1973) 37-74.

[12] R. Pöschel, Die Anzahl der maximalen superpositionsabeschlossenen Klassen von Funktionen über einer endlichen Familie endlicher Mengen. Preprint 1974.

[13] E. Post, *Two Valued Iterative Systems of Mathematical Logic*, Annals of Math. Studies 5, Princeton 1941.

[14] I. Rosenberg, La structure des fonctions de plusieurs variables sur un ensemble fini, *C.R. Acad. Sci. Paris* Ser A-B 260 (1965) 3817-3819.

[15] I. Rosenberg, Über die funktionale Vollständigkeit in der mehrwertigen Logiken, *Rozpravy Československé Akad. Věd.*, Řada Mat. Přir, Věd. 80, 4 (1970) 1-93.

[16] I. Rosenberg, Algebren und Relationen, *Elektronische Informations-verarbeitung und Kybernetik* 6, 2 (1970) 115-124.

[17] I.G. Rosenberg, Une correspondance de Galois entre les algèbres universelles et les relations dans le même univers, *C.R. Acad. Sci. Paris* (1975).

[18] I.G. Rosenberg, A Galois connection between universal algebras and relations, preprint *Publication du Centre de Recherches Mathématiques* no. 393, Université de Montréal.

[19] I.G. Rosenberg, Completeness, closed classes and relations in multiple-valued logics. *Proceedings of the* 1974 *Inter. Symp. on Multiple-valued Logic*, 1-26.

[20] J. Słupecki, Completeness criterion in many-valued logical systems, *Comptes rendus des séances de la Soc. des Sci. et les Lettres de Varsovie* Cl. III 32 (1939) 102-109, English translation *Studia logica* 30 (1972) 153-157.

[21] V.V. Martynjuk, Investigation of some classes of functions in many-valued logics (Russian), *Problemy kibernetiki* 3 (1960) 49-60.

THE LINEARITY PROPERTY AND FUNCTIONAL COMPLETENESS IN M-VALUED LOGIC

Hamed A. Ellozy
Computer Sciences Department
IBM Thomas J. Watson Research Center
Yorktown Heights, New York 10598

Yale N. Patt
Computer Science Department
NC State University
Raleigh, N.C. 27607

Introduction

In previous papers, Post (1), Webb (2) and Martin (3) among others presented some Sheffer functions in M-valued logic.

Martin (4) established some general properties of Sheffer functions and isolated all the Sheffer functions in 3-valued logic.

Patt (5) defined a new property, linearity, and proved that nonlinearity is a necessary condition for logical completeness in M-valued logic.

In the present paper, we will show that Martin's four properties reduce to three, determine the properties of linear functions and establish a new property, P', necessary for completeness in M-valued logic.

Martin's Four Properties

Martin (4) has shown that a necessary and sufficient condition for a function of two variables to be complete in three-valued logic is that this function possesses none of the following four properties:

(a) the proper substitution property,
(b) the co-substitution property,
(c) the proper closing property, and
(d) the t-closing property.

We first need to define a few terms that will be used subsequently. Definitions 1 to 9 are due to Martin (4).

Definition 1. If i and j are truth values ($0 \leq i, j \leq M - 1$) and if D is a decomposition (partition) of the truth values (0,...,M-1) into two or more disjoint subclasses then we will say $i \sim j(D)$, read as i and j are in the same class, if i and j are elements of the same class.

Definition 2. A two variable function f(x,y) satisfies the *substitution law* for a decomposition D if for any truth values h, i, j, and k, whenever $h \sim j(D)$ and $i \sim k(D)$ then $f(h,i) \sim f(j,k)(D)$.

Definition 3. A two variable function f(x,y) satisfies the *co-substitution law* for D, if for any values h, i, j, and k whenever $f(h,i) \sim f(j,k)(D)$ then $h \sim j(D)$ or $i \sim k(D)$.

Definition 4. A two variable function f(x,y) has the *proper substitution property* if there is a decomposition of the truth values in less than M classes for which it satisfies the substitution law.

Definition 5. A two variable function f(x,y) has the *co-substitution property* if there is a decomposition of the

This paper was supported in part by AROD Grant 31-124-G186 and is based on the first author's Ph.D. dissertation submitted to the Electrical Engineering Department at NC State University, May 1973.

truth values for which is satisfies the co-substitution law.

Definition 6. A one variable function f(x) is a *t-function* t(x) if $t^M(j) = j$ for all $j(0 \leq j \leq M - 1)$ and for every i and j $(1 \leq i \leq M\text{-}1; 0 \leq j \leq M\text{-}1)$, $t^i(j) \neq j$.

Definition 7. A two variable function f(x,y) is said to be *t-closing* if there is a t function t(x) such that for every i and j there is a k such that $f(t^i(x), t^j(x) = t^k(x)$.

Definition 8. A two variable function f(x,y) is *properly closing* if some nonempty proper subset of the M truth values is closed under f(x,y). Martin (6) proved that no function f(x,y) when possessed one or more of these properties could be a Sheffer function. He showed that a Sheffer function must have the property P which he defines as follows.

Definition 9. A two variable function f(x,y) has property P if it has none of the following properties: proper substitution, co-substitution, proper closing and t-closing.

Finally Martin proved that, in three-valued logic, a function of two variables f(x,y) is a Sheffer function iff it has property P.

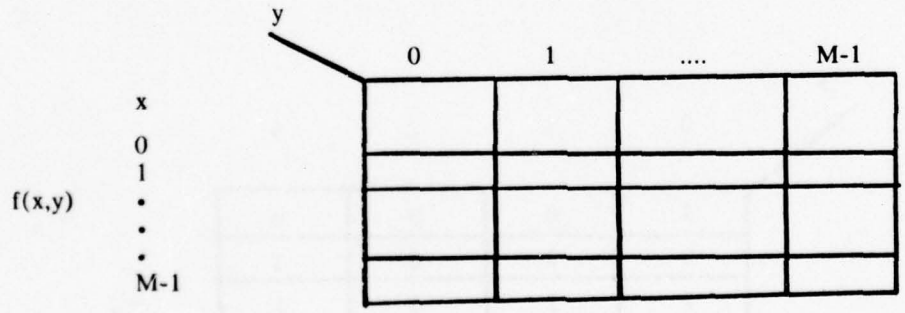
We will now proceed to show that Martin's four properties reduce to three.

Reduction of Martin's Four Properties to Three

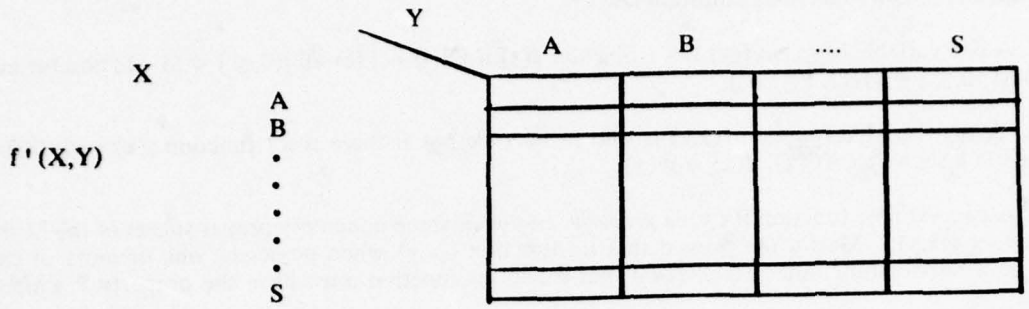
Patt and Cooper (6) have shown that if a function f(x,y) has the co-substitution property then f(x,y) has the proper substitution property.

Let the two-dimensional space of the function f(x,y) be partitioned into S regions, where S is the number of subsets in the partition of the truth values (D). The truth values in each region belong to only one subset of (D). We shall show that when the truth values are partitioned into M subsets, the co-substitution property implies that f(x,y) is a function of one variable.

Proof. Represent the function f(x,y) by an M × M map.



Patt and Cooper (6) have shown that if a function satisfies the co-substitution low, then it also satisfies the proper substitution law. We can relabel the truth values so that the members of each subset will be adjacent to one another in the relabeled coordinates. Denote the subsets by capital letters then f ' (x,y) can be represented as:



The entries in each region of the map must be from the same subset (proper substitution). Let the entries in the region whose coordinates are (A,A) be from the subset $\alpha(\alpha \in \{A,B,...,S\})$. From the definition of the co-substitution property, the only other regions in the map that could be filled in with entries from α are either the regions whose X coordinate is A or those whose Y coordinates is A. Furthermore, they are mutually exclusive, i.e., we cannot have elements from α in the two regions whose coordinates are (A,B) and (B,A).

Assume that the other region, whose entries are elements of α, has its X coordinate give by A. The only values that appear along row A must all be from α, as any other value would eventually result in a conflict. In the same way, the values of the entries along row B must be from the set $\beta(\beta \neq \alpha)$..., etc., i.e., to each row is associated one particular set from the partition, and no set is associated with more than one row. By the same token, f ' (X,Y) could have been partitioned along its column instead of its rows.

When the truth values are partitioned in M nonempty subsets, i.e., each subset consists of exactly one element, then f(x,y) reduces to a function of one variable if f(x,y) has the co-substitution property.

Therefore, if a function f(x,y) in M-valued logic has the co-substitution property, then it is either a function of one variable, i.e., reducible or, if it is irreducible, it has the proper substitution property. We can thus redefine the property P as follows.

Definition 10. An irreducible function f(x,y) in M-valued logic, has property P if it has none of the following properties: proper closing, proper substitution, and t-closing.

We now proceed to give an example of a four-valued function f(x,y) that has the property P and yet is not complete.

y

x \ y	0	1	2	3
0	2	0	3	0
1	0	2	0	3
2	1	2	0	2
3	2	1	2	0

f(x,y)

It can be readily shown that this function possesses the property P and it can also be shown that this function is not complete. It turns out that this function is a linear function.

The Linearity Property

Definition 11. A function f(x,y) is linear with index α if there exists a value $\alpha \in \{0,1,...,M-1\}$ such that $f(x + \alpha, y + \alpha) = f(x,y) + \alpha$ where the addition is mod M. Conversely if there exists no α such that f satisfies the linearity requirement for all x and y, then f is *nonlinear* (5).

Lemma 1. Nonlinearity is a necessary condition for logical completeness in M-valued logic [Patt (5)].

It should be pointed out that the completeness property is unaffected by a relabeling of the M truth values. If a function is complete is must generate all the functions of any number of variables. If we now relabel the truth values of that function, the resulting function is isomorphic to the first function, i.e., if the initial function f generates a function $h(x) = a_x$, and if the relabeling is of the form $P_0,P_1,....,P_{M-1}$, then if we denote the relabeling of f and f ', f ' generates a function h '(x) such that $h'(P_x) = P_{a_x}$ and therefore if f(x) is complete then f '(x) is also complete.

It should be noted that Martin's properties are invariant under relabeling whereas linearity is not. Martin's properties are associated with partitions of the truth values, i.e., set properties and these are always invariant whereas the linearity property is an algebraic property and is not preserved.

As an example consider the function f(x,y) defined in the preceding figure. This function is linear. If we relabel the values of the function according to the following scheme: 0 is replaced by 0, 1 by 2, 2 by 1, and 3 by 3 we obtain the following function.

f '(x,y)

x \ y	0	1	2	3
0	1	3	0	0
1	2	0	1	1
2	0	0	1	3
3	1	1	2	0

f '(x,y) is not linear.

Definition 12. A function f(x,y) in M-valued logic has the *implied linearity* property if there exists a relabeling of the truth values that transforms f(x,y) into a linear function.

We shall show later on how to test a function for implied linearity. We now proceed to prove the following lemmas and theorems as they will be useful in the determination of implied linearity.

Theorem 1. If a function f(x,y) is linear with index α then f(x,y) is linear with index β where β is the largest common factor of α and M.

Proof. Express α as $\alpha = i\beta$ and M as $M = j\beta$,i and j have no factor other than unity in common. Now, $\alpha j = ij\beta = iM = 0$ and mod M. f(x,y) being linear, the sequence of function values obtained starting from an arbitrary (x_o, y_o) is pair

$f(x_o, y_o) = a$
$f(x_o + \alpha, y_o + \alpha) = a + \alpha$
$f(x_o + (j - 1)\alpha, y_o + (j-1)\alpha) = a + (j-1)\alpha$
$f(x_o + j\alpha, y_o + j\alpha) = f(x_o, y_o) + j\alpha = a + 0 = a.$

Thus we see that an arbitrary (x_o, y_o) pair generates exactly j different values.

This is easily verified as follows. The upper bound on the number of distinct values is j. Let us assume for the moment that less than j distinct values are generated, i.e., there exist two values l and $m < j$ such that $l\alpha = m\alpha$ and $l \neq m$:

$(l\text{-}m)\ \alpha = 0 \bmod M.$

Recall that $\alpha = i\beta$, $M = j\beta$, and j and i have no factors in common. $(l\text{-}m)\alpha$ can be rewritten as $(l\text{-}m)i\beta = 0$.

Since i has no factors in common with M either

(i) $l\text{-}m = 0 \bmod M$ implying that $l = m$ (both l and $m < M$) and contradicting the original assumption that $l \neq m$ or

(ii) $(l\text{-}m)\beta = M$ implying that $l\text{-}m = j$.

Since both l and m are less than j, their difference cannot equal j.

Hence no two distinct values l, $m < j$ exist such that $l\alpha = m\alpha$. Thus an arbitrary (x_o, y_o) pair generates exactly j different function values. The rest of the function values are independent of (x_o, y_o). The j different (x,y) pairs that depend upon the (x_o, y_o) pairs are $(x_o, y_o + \alpha)$, $(x_o + 2\alpha, y_o + 2\alpha)$,...,$(x_o + (j\text{-}1)\alpha, y_o + (j\text{-}1)\alpha)$, and (x_o, y_o). We will now show that if $f(0,0) = 0$ then $f(k\beta,k\beta) = k\beta\ \forall k$.

Since we have j different function values for the j(x,y) pairs defined above, and since the function values are cyclic modulo M, then these values must occur at equal intervals β over the interval $\{0,1,...,M\text{-}1\}$ since $j\beta = M$.

Therefore if $f(0,0) = 0$ then $f(\beta,\beta) = \beta$, $f(2\beta, 2\beta) = 2\beta$,..., $f[(j\text{-}1)]\beta, (j\text{-}1)\beta] = (j\text{-}1)\beta$.

Now if $f(0,0)$ was $= a$ instead of 0, this would amount to adding the value a to each function values; the intervals β between the function values are unchanged, i.e., if $f(0,0) = a$ then $f(\beta,\beta) = a + \beta$, $f(2\beta, 2\beta) = a + 2\beta$,...,$f[(j\text{-}1)\beta, (j\text{-}1)\beta] = a + (j\text{-}1)\beta$.

By the same token if we let $x = x_o$ instead of 0 and $y = y_o$ instead of 0 we obtain $f(x_o, y_o) = a$,$f(x_o + \beta, y_o + \beta) = a + \beta$,..., $f(x_o = (j\text{-}1)\beta, y_o + (j\text{-}1)\beta) = a+(j\text{-}1)\beta$, which means that f(x,y) is linear with index β.

Corollary 1. If a function f(x,y) is linear with index α, then f(x,y) is linear with index 1 if α and M are relatively prime.

Proof. The proof follows directly from theorem 1. When α and M are relatively prime then $\beta = 1$ implying that f(x,y) is linear with index 1. Q.E.D.

Lemma 2. If a function f(x,y) is linear with index β and there exists no $\gamma < \beta$ such that f(x,y) is linear with index γ, then given an arbitrary (x_o, y_o) pair, the sequence of function values $f(x_o + i, y_o + i)$ contains exactly β independent function values.

Proof. f(x,y) is of the form

$$f(x_o, y_o) = a_0$$
$$f(x_o + 1, y_o + 1) = a_1$$
$$\vdots$$
$$f(x_o + \beta - 1) = a_{\beta-1}$$
$$f(x_o + \beta, y_o + \beta) = a_0 + \beta$$
$$f(x_o + \beta + 1, y_o + \beta + 1) = a_1 + \beta$$

etc., i.e., when the input values range form (x_o, y_o) to $(x_o + \beta - 1, y_o = \beta + 1)$, all the function values are independent from one another and outside that range all the other function values depend upon the values in the range (x_o, y_o) to $(x_o + \beta\text{-}1, y_o + \beta - 1)$.

Lemma 3. If a function f(x,y) is linear with index α, then f(x,y) is linear with index $k\alpha$ for all k.

Proof. If f(x,y) is linear with index α then

$$f(x_o + \alpha, y_o + \alpha) = f(x_o, y_o) + \alpha.$$

It follows that

$$f(x_o + 2\alpha, y_o + 2\alpha) = f(x_o + \alpha, y_o + \alpha) + \alpha = f(x_o, y_o) + 2\alpha.$$

Repeating this procedure iteratively we obtain

$$f(x_o + k\alpha, y_o + k\alpha) = f(x_o, y_o) + k\alpha$$

for all (x_o, y_o) pairs.

From the definition of linearity it follows that f(x,y) is linear with index $\alpha' = k\alpha$. Q.E.D.

Corollary 2. If a function f(x,y) is linear with index α, where α and M are relatively prime, then f(x,y) is linear with index k for all k.

Proof. If α and M are relatively prime then f(x,y) is linear with index 1 (Corollary 1). From lemma 3 if f(x,y) is linear with index 1 it is also linear with index $k \times 1 = k$ for all k. Q.E.D.

Thus any linear function with index α, α and M being relatively prime, can be expressed as a linear function with index β where β is any factor of M.

Lemma 4. Any cyclic relabeling does not affect the linearity property.

Proof. Let the relabeling be of the form j to j + k mod M $\forall$ j. f(x,y) is of the from

$$\begin{aligned} f(x_o + i, y_o + i) &= a_i \quad \forall i\ 0 \le i \le \beta\text{-}1 \\ &= a_{i-\beta+\beta} \quad \forall i\ \beta \le i \le 2\beta\text{-}1 \quad \text{etc.} \end{aligned}$$

Under the relabeling of the truth values f becomes

$$\begin{aligned} f(x_o + k + i, y_o + k + i) &= a_i + k \quad \forall i\ 0 \le i \le \beta\text{-}1 \\ &= a_{i-\beta+\beta+k} \quad \forall i\ \beta \le i \le 2\beta\text{-}1 \quad \text{etc.} \end{aligned}$$

But in our definition of a linear function, (x_o, y_o) was arbitrary and so were the values a_i. Thus if we let $x_o + k = x_o'$, $y_o + k = y_o'$, and $a_i + k = a_i'$, then f(x,y) becomes

$$\begin{aligned} f(x_o', y_o') &= a_i' \quad \forall i\ 0 \le i \le \beta\text{-}1 \\ &= a'_{i-\beta+\beta} \quad \forall i\ \beta \le i \le 2\beta\text{-}1 \quad \text{etc.} \end{aligned}$$

i.e., f'(x,y) is linear with index β. Furthermore, if f(x,y) is nonlinear to start with, it remains nonlinear under any

cyclic relabeling. Q.E.D.

We have shown in lemma 4 that the linearity property is invariant under cyclic relabeling, i.e., if a function f(x,y) is linear (nonlinear) under a relabeling of the form

0	1	2	...	M-1
a_o	a_1	a_2	...	a_{n-1}

then it is also linear (nonlinear) under the relabelings $(a_o + i, a_1 + i,...,a_{M-1} + i)$. We can now state the following theorem.

Theorem 2. To determine whether a function does or does not possess the implied linearity property, (M-1)! relabeling of the truth values are sufficient.

Proof. The total number of relabeling of the truth values is M!. These M! relabelings can be partitioned into (M-1)! equivalence classes, the members in each class being $(a_o + i, a_1 + i,...,a_{M-1} + i)$ for all i.

Since the members of a given class are either all linear or all non-linear (lemma 4), then to determine whether or not a function has the implied linearity property, one need only relabel the function at most (M-1)! times, one relabeling for each class being used each time. Q.E.D.

We can now determine whether a function has the implied linearity property as follows.

Corollary 3. A nonlinear function f(x,y) in M-valued logic has the implied linearity property iff there exists a least one relabeling among the [(M-1)!-1] relabelings of its truth values such that the relabeled function f(x,y) is linear with index β. (β is any factor of M.)

Proof. Let the (M-1)! relabelings be of the form

0	1	2	...	M-1
0	a_1	a_2	...	a_{M-1}

where the sets $\{a_i\}$ are the (M-1)! permutations of the truth values $1 \leq i \leq M - 1$. These (M-1)! relabelings are sufficient to determine if the function has the implied linearity property (theorem 4.2). From these relabelings we can remove the identity relabeling as it corresponds to the cyclic relabeling. [f(x,y) is assumed to be nonlinear, and we have shown in lemma 4 that the linearity property is invariant under cyclic relabeling.]

Thus if f(x,y) is not linear under any of these relabelings, then f(x,y) does not possess the implied linearity property. Q.E.D.

Remark. Patt (5) has shown that if M is a prime number, then the linearity property and the t-closing property are equivalent. Thus in the three-valued case, any linear function is also t-closing and vice-versa.

Definition 13. The *diagonal* of a function f(x,y) is the set {f(i,i)} for all i.

To establish implied linearity in M-valued logic we proceed as follows:

(1) First determine from the diagonal whether the function may be implied-linear. Check the diagonal values according to the previous criterion.

(2) If the answer is positive, determine whether any of the (M-1)!-1 relabeled functions is linear.

We will now show that the t-closing property is a special case of the implied-linearity properity. Recall that a one variable function f(x) is a t-function if.

$$t^i(x) \neq x \qquad \forall x, \forall i \quad 1 \leq i \leq M-1$$

and

$$t^M(x) = x.$$

It following from the definition that a t-function can be represented by

$t(\alpha_i) = \alpha_{i+1}$ where $\alpha_i \in \{0,1,...,M-1\}$ and $\alpha_i \neq \alpha_j$.

We can determine the set $\{\alpha_i\}$ as follows:

let

$$\alpha_0 \triangleq 0$$

then

$$\alpha_1 = t(\alpha_0)$$

$$\alpha_2 = t(\alpha_1) \quad \text{etc.}$$

It will also be recalled (4) that a function f(x,y) is t-closing if there exists a t-function such that for every i and j there exists a k such that

$$f(t^i(x), t^j(x)) = t^k(x).$$

This is equivalent to $f(t^i(\alpha_x), t^j(\alpha_x)) = t^k(\alpha_x)$ therefore

$$f(\alpha_{x+i}, \alpha_{x+j}) = \alpha_{x+k}.$$

Relabel the values $\{\alpha_x\}$ so that the new value of each α_x is equal to x then f becomes

$$f'(x + i, x + j) = x + k \quad \forall x.$$

Let $x_o = i$, $y_o = j$, and $a = k$. Then $f'(x_o + x, y_o + x) = a + x \quad \forall x$ which means that

$$f'(x_o, y_o) = a$$
$$f'(x_o + 1, y_o + 1) = a + 1$$
$$f'(x_o + 2, y_o + 2) = a + 2 \quad \text{etc.}$$

i.e., f'(x,y) is linear with index 1.

Definition 14. An irreducible function f(x,y) in M-valued logic has property P' if it has none of the following properties: proper closing, proper substitution, and implied linearity.

We can now state the following extension of Martin's Theorem V(4).

Theorem 3. Every Sheffer function possesses the property P'.

Proof. The proof follows directly from Martin's theorem V(4) and Lemma 1.

Conclusions

We would like to summarize what has been presented in this paper Marin's properties were shown to be insufficient for logical completeness for M > 3. Patt and Cooper's proof that the co-substitution property implies the proper substitution was extended. A new property, implied linearity, was introduced and it was shown that one of Martin's properties, the t-closing property, is a special case of the implied linearity property.

Finally, a new property necessary for completeness in M-valued logic, the P ' property, was defined.

REFERENCES

[1] E. L. Post, "Introduction to a General Theory of Elementary Propositions", *American J. of Math.*, Vol. 43, (1921) pp. 163-185.

[2] D. L. Webb, "Generation of any N-valued Logic by One Binary Operation", Proc. of the Nat. Academy of Sci., Vol. 21 (1953), pp. 102-109.

[3] N. M. Martin, "Sheffer Functions and Axiom Sets in M-Valued Propositional Logic", Ph.D. Thesis, UCLA, (1952).

[4] N. M. Martin, "The Sheffer Functions of 3-valued Logic", *J. of Symbolic Logic,* Vol. 19 (1) (1954), pp. 45-51.

[5] Y. N. Patt, "Nonlinearity: A New Necessary Condition for Logical Completeness in K-valued Logic", Proc. of the Sixth Ann. Hawaii Conf. on System Sci., University of Hawaii, Honolulu, Hawaii (1973).

[6] Y. N. Patt, G. A. Cooper, "Toward a Characterization of Logically Complete Switching Functions in K-valued Logic", Conference Record of the 1972 Symposium on the Theory of an Application of Multiple-Valued Logic Design, State University of New York, Buffalo, New York (1972).

SECOND ORDER AND HIGHER ORDER UNIVERSAL DECISION ELEMENTS IN m-VALUED LOGIC

John Loader
Brighton Polytechnic
Brighton, England

ABSTRACT

Second order and higher order universal decision elements (universal logic modules) are discussed and some lower bounds for the number of inputs are found. The 3-valued case is considered and it is shown that an 8 input functor corresponding to a universal decision element exists.

INTRODUCTION

An Nth order universal decision element of the rth degree in m-valued logic may be defined as follows:

A functor $\Phi(\ ,\ldots,\ )$ of n argument places corresponds to an Nth order universal decision element of the rth degree if we may define all non-trivial functors of m-valued logic with N or less argument places, solely by substitution of the formulae $F_1(P_1),\ldots,F_r(P_1),\ldots,F_1(P_n),\ldots,F_r(P_n)$, $(F_1(P)=_T P)$, and the logical constants $1,\ldots,m$ in its argument places, the functor $\Phi(\ ,\ldots,\ )$ being used only once in the definiens.

For 2-valued logic the case N=2, r=1 has been considered in detail by Sobociński [1] and Foxley [2]. This has been extended to higher values of N by a number of authors (see, for example, [3], [4]). For m-valued logic Rose [5] and Loader [6], [7], [8] have considered the case N=1 with various values of r.

In this paper we will consider second and higher order universal decision elements of the first degree and show that in 3-valued logic there exists an 8 input functor that is suitable. Throughout we will denote the m values of m-valued logic by the integers $1,\ldots,m$. The entry points in the truth table of the formula $\Phi(P_1,\ldots,P_n)$ will be numbered according to the rule:

$$\text{entry point number, } j = \sum_{k=1}^{n} (x_k-1).m^{n-k} + 1 \ ,$$

where x_k is the value taken by the variable P_k, $k=1,\ldots,m$.

LOWER BOUNDS

If $\Phi(P_1,\ldots,P_n)$ corresponds to an (N-1)th order universal decision element for m-valued logic we may immediately deduce the formula $\Phi_1(P_1,\ldots,P_{mn+1})$ of mn+1

argument places which corresponds to an Nth order universal decision element as follows:

$$\Phi_1(P_1,\ldots,P_{mn+1}) =_T \left[P_1,\ \Phi(P_2,\ldots,P_{n+1}),\ldots,\ \Phi(P_{n(m-1)+2},\ldots,P_{mn+1}),\ P_1\right],$$

where $[\ ,\ldots,\ ]$, the generalised conditioned disjunction functor, is such that $[P,Q_1,\ldots,Q_m,P]$ takes the truth value of Q_i when P takes the truth value $i, i=1,\ldots,m$. Thus for the cases m=3 and m=4 where we know that first order universal decision elements of m argument places exist [7], we may deduce second order universal decision elements of 10 and 17 argument places respectively and hence third order ones of 31 and 69 argument places.

For the second order case we may deduce a lower bound for the number of argument places required, using an argument similar to that used by Sobociński [1] for the 2-valued case. First we make the following definition:

A binary functor will be said to be trivial if it satisfies one of the following conditions:

(i) all rows or all columns of its corresponding truth table are identical;

(ii) if $\Lambda_1(P,Q) =_T \Lambda_2(Q,P)$ where $\Lambda_1(\ ,\ )$ and $\Lambda_2(\ ,\ )$ are distinct functors, then one of $\Lambda_1(\ ,\ )$, $\Lambda_2(\ ,\ )$ is said to be trivial.

From this definition it is easily shown that the number of non-trivial binary functors in m-valued logic is given by

$$\tfrac{1}{2}m^m(M^2 + M - 2), \text{ where } M = m^{m(m-1)/2}.$$

For m=2 and m=3 the above expression gives the values 8 and 10179 respectively.

Theorem 1. If $\Phi(\ ,\ldots,\ )$ is a functor of n argument places of m-valued logic and if n is such that the inequality

$$(m+2)^n - 2(m+1)^n + m^n - 3^n + 2^{n+1} - 1 < (m^{m^2} + m^{m(m+1)/2} - 2m^m)(m-1)/m$$

holds then $\Phi(\ ,\ldots,\ )$ cannot correspond to a second order universal decision element.

Proof. It is easily shown that the number of different ways in which the argument places of the functor $\Phi(\ ,\ldots,\ )$ may be filled from the set $\{P,Q,1,\ldots,m\}$ such that at least one P and one Q occur is given by $(m+2)^n - 2(m+1)^n + m^n$. Of these possible substitutions $3^n - 2^{n+1} + 1$ will use entry point 1 since these substitutions must be such that the argument places are filled from the set $\{P,Q,1\}$. Thus the number of substitutions which do not use entry point 1 is given by

$$(m+2)^n - 2(m+1)^n + m^n - 3^n + 2^{n+1} - 1.$$

Now suppose that $\Phi(P_1,\ldots,P_n)$ takes the truth value k, $k\varepsilon\{1,\ldots,m\}$, when $P_1,\ldots,P_n$ all take the truth value 1. Then in order to define those binary functors which have the first entry in their corresponding truth table different from k, we must use only those substitutions which do not utilise entry point 1. Now the number of symmetrical truth tables where the first entry is not k is given by $(m^{m(m+1)/2} - m)(m-1)/m$ and the number of remaining non-trivial binary functors where

the first entry in the corresponding truth table is not k is given by

$$\tfrac{1}{2}(m^{m^2} - m^{m(m+1)/2} - 2m^m + 2m)(m-1)/m \quad .$$

Thus to define each of these we must use a substitution where entry point 1 is not utilised. Further, there must exist two such substitutions since if the truth table is symmetric we must be able to interchange P and Q without upsetting the definition, and if the truth table is non-symmetric there will be a corresponding trivial functor which must be definable by interchanging P and Q.

Thus the number of such substitutions required is given by

$$2(m^{m(m+1)/2} - m)(m-1)/m + (m^{m^2} - m^{m(m+1)/2} - 2m^m + 2m)(m-1)/m$$

i.e. $(m^{m^2} + m^{m(m+1)/2} - 2m^m)(m-1)/m$.

But from the above the number of substitutions available is $(m+2)^n - 2(m+1)^n + m^n - 3^n + 2^{n+1} - 1$ and hence if the inequality stated in the theorem holds, $\Phi(\ ,\ldots,\ )$ cannot correspond to a second order universal decision element.

The table below shows the minimum value of n where the inequality of theorem 1 breaks down, for m=2,...,10

m	2	3	4	5	6	7	8	9	10
minimum n	4	7	13	21	31	44	58	75	93

A less complicated method of finding a lower bound for n, for any N, may be achieved by simply considering the total number of substitutions available and the total number of functors of N argument places to be defined (including the trivial ones). We first prove the following theorem.

Theorem 2. If $\Phi(\ ,\ldots,\ )$ is a functor of n argument places then the number of ways of substituting in the argument places of $\Phi(\ ,\ldots,\ )$ from the set $\{P_1,..,P_N,1,\ldots,m\}$ (N<m) such that each P_i (i=1,...,N) occurs at least once is given by

$$\rho(N) = \sum_{j=0}^{N} (-1)^j \binom{N}{j} (m+N-j)^n \quad .$$

Proof. The proof is by strong induction on N. If N=1 then the number of substitutions is $(m+1)^n - m^n$ and this may be expressed in the form

$$\sum_{j=0}^{1} (-1)^j \binom{1}{j} (m+1-j)^n = \rho(1) \quad .$$

Now suppose that the result holds for N=1,...,r and consider the case N=r+1. The substitution set is $\{P_1,\ldots,P_{r+1},1,\ldots,m\}$ and the total number of ways of substituting in the argument places of the functor $\Phi(\ ,\ldots,\ )$ without restriction is $(m+r+1)^n$. Now the number of substitutions such that each P_i (i=1,...,r+1) occurs atleast once may be expressed as the total number of substitutions without restriction less the number of

substitutions such that exactly k of the P_i occur at least once for all k=0,...,r . On the induction hypothesis the number of substitutions with any k of the P_i occurring at least once is given by $\binom{r+1}{k}\rho(k)$. Thus the required number of substitutions is given by

$$(m+r+1)^n - \sum_{k=0}^{r}\binom{r+1}{k}\sum_{j=0}^{k}(-1)^j\binom{k}{j}(m+k-j)^n \ .$$

Rearranging the terms in the double summation this may be written

$$(m+r+1)^n + \sum_{j=1}^{r+1}(-1)^j\binom{r+1}{j}(m+r+1-j)^n$$

$$= \sum_{j=0}^{r+1}(-1)^j\binom{r+1}{j}(m+r+1-j)^n \quad = \quad \rho(r+1) \ .$$

Hence the result is proved.

From theorem 2 we may deduce that $\Phi(\ ,\ldots,\ )$ cannot correspond to an Nth order universal decision element for m-valued logic if

$$\rho(N) < m^{m^N} \ .$$

The table below shows the minimum value of n where this inequality breaks down for m=2,...,10 ; N=1,...,5.

m \ N	1	2	3	4	5
2	2	3	5	8	12
3	3	7	17	46	129
4	4	13	46	171	647
5	5	21	97	458	2185
6	6	32	177	1009	5811
7	7	44	290	1949	13162
8	8	58	445	3428	26566
9	9	75	645	5621	49163
10	10	93	898	8726	85028

THE 3-VALUED CASE

From above we know that if $\Phi(P_1,\ldots,P_n)$ corresponds to a second order universal decision element for 3-valued logic then $n\geq 7$. The number of different ways in which the argument places of the formula $\Phi(P_1,\ldots,P_n)$ may be filled from the set {P,Q,1,2,3} such that at least one P and one Q occur is given by $5^n-2.4^n+3^n$, which gives the value 47544 when n=7.

Loader [7] describes a general method for finding universal decision elements and this method was adopted starting with an arbitrary formula $\Phi(P_1,\ldots,P_7)$. Initially

the number of binary functors undefined was found to be 2070. After considering 1281 entry points the formula $\Phi_1(P_1,\ldots,P_7)$ was found where the number of undefined binary functors was 1055. However, this was achieved at the expense of over 400 hours machine time using an IBM 1130 configuration. At this stage an attempt was made to find a formula of 8 argument places corresponding to a universal decision element. Initially the formula

$$\Lambda(P_1,\ldots,P_8) =_T \left[P_1,\ \Phi_1(P_2,\ldots,P_8), \sim\Phi_1(P_2,\ldots,P_8),\ \sim\sim\Phi_1(P_2,\ldots,P_8),\ P_1\right]$$

was considered where $\sim$ corresponds to the cyclic negation functor of Post [9]. The number of undefined binary functors was found to be 12 and proceeding with the method the formula

$$\Lambda_1(P_1,\ldots,P_8) =_T \left[P_1,\ \Phi_2(P_2,\ldots,P_8), \sim\Phi_2(P_2,\ldots,P_8),\ \sim\sim\Phi_2(P_2,\ldots,P_8),\ P_1\right]$$

corresponding to a second order universal decision element was found.

REFERENCES

[1] Sobociński,B. — On a universal decision element. J.Comput.Systems 1(1953), 71-80.

[2] Foxley,E. — Determination of the set of all four variable formulae corresponding to universal decision elements using a logical computer. Zeitschr.f.math.logik und Grundlagen d.Math., 10(1964),302-314.

[3] King,W.F.,III — The synthesis of multi-purpose logic devices. Seventh Annual Symposium Switching and Automata Theory, Conf.Rec.(1966), 227-235.

[4] Preparata,F.P. and Muller,D.E. — Generation of near optimal universal Boolean functions. J.Computer Syst.Sci., 4(1970), 93-102.

[5] Rose,A. — Sur les éléments universal trivalent de décision. Comptes Rendus, 269(1969), 1-3.

[6] Loader,J. — Universal decision elements in m-valued logic. Zeitschr.f.math.logik und Grundlagen d.Math.,18(1972),205-216

[7] Loader,J. — A method for finding formulae corresponding to first order universal decision elements in m-valued logic. Zeitschr.f.math.logik und Grundlagen d.Math.,20(1974),1-18.

[8] Loader,J. — An alternative concept of the universal decision element in m-valued logic. Zeitschr.f.math.logik und Grundlagen d.Math., to appear 1975.

[9] Post,E.L. — Introduction to a general theory of elementary propositions. American J. of Math., 43(1921),163-185.

THE LOGICAL FOUNDATIONS OF

MICROLANGUAGES

T.C. Wesselkamper
Virginia Polytechnic Institute
and State University
U.S.A.

I. The Problem

A. Electronic digital computers began in a world of vacuum tubes and diodes. They were endowed with sets of machine instructions heavily influenced by the available electronics of the time. It is not possible that anything else could have occurred.

In nearly three decades since, technology has changed mightily. Machines have become word addressable: their electronic componenets moves through transistor technology into integrated circuit technology. Sets of machine instructions have grown: IBM's System/370 features over one hundred fifty instructions.

Recent years have witnessed a growing interest in microprogramming. Often this represents a return to the consciousness that there is a machine beneath all those layers of high level language.

This interest in microprogramming and the correlative ability to easily emulate machines in what has come to be called firmware makes it possible to ask a set of questions which (logically) should have been asked before the first machine was constructed. It is now possible to ask: What should a machine be able to do?

B. Two recent examples exhibit the current state of the pragmatic answer to this question. The first is the Weisbecker machine [1]. In an excellent paper Joe Weisbecker "describes a simplified microcomputer architecture that offers maximum flexibility at minimum cost." [1, p. 41] We are told:

> "The ALU is an 8-bit logic network for performing binary add, subtract, logical 'and', 'or', and 'exclusive or' on two 8-bit operands. One operand is the bus byte and the other is contained in the D register. The D register can also be shifted right one bit position. Add, subtract, and shift operations set a one bit overflow register...which can be tested by a branch instruction." [1, p. 43]

No attempt is made to explain this choice of functions. They provide a typical example of the operations provided by designers.

The second recent example is a description of HALL [2], an assembler level language for the HYRMAN hardware simulator [3]. The HALL machine possesses nine arithmetic functions (binary and decimal addition and subtraction, 'and', 'or',

'exclusive or', left shift and right shift) and fourteen status instructions. These are given in Table I, below. (It should be noted that HALL does decimal arithmetic in a fashion analogous to the old IBM 1620 or to the S/360-370 "packed decimal" arithmetic).

That these designs have not varied significantly in thirty years may be seen by comparing them to the instructions proposed by John von Neumann for the EDVAC machine in 1945. [11] See Table II, below. This in spite of the dual facts that electronics can support much more varied design and that the class of problems to which computers have come to be applied is far wider than was envisioned when the first computers were designed for numeric work.

II. General Purpose Computers and Complete Sets of Functions

A. This paper examines some of the possible answers to the question: What should a machine be able to do? We are concerned that while one hundred fifty instructions are quite likely to be sufficient for any job, they are not likely to be necessary. Even if one argues that they are desirable at some level of machine definition, they are not desirable at the microlevel.

We assume that when a computer manufacturer says that his machine is a "general purpose machine" he means that it is functionally complete over the space of its words.

More specifically, we assume a hypothetical machine is fixed word size, not necessarily binary. The value of a word ranges over a set $E(k) = \{0, 1, \ldots, k-1\}$. A machine M with words in E(k) is complete if whenever n is a natural number and $f: E^n(k) \to E(k)$ is a function over E(k) then for any set of values $(x_1, x_2, \ldots, x_n)$ it is possible to evaluate $f(x_1, x_2, \ldots, x_n)$ on M.

This notion of completeness is implied by, but not necessarily equivalent to, the completeness with constants of the set A of those machine operations which have the property of being functions over E(k). This is opposed to those machine operators which are control operators, for example, a branch instruction. In Section III of this paper we survey some of the known results about complete sets of functions.

A multiple-valued logician succumbs to the temptation to regard a general purpose computer as a function evaluator. The remarkable advances in the last twenty years in the theory of complete sets of functions make it possible to make exact and general statements about independent, complete sets of operators. As a result of the work of recent years, particularly that of Professor Rosenberg, it is now possible to identify which sets of operators are sufficient to be the fundamental instruction set of a machine. The economic goal of minimizing cost leads to a concern with independence; the "general purpose" criterion dictates completeness.

B. In the practical area of machine design two other questions arise in the process of evaluating a proposed instruction set:

(1) Can normal human beings (as opposed to multiple-valued logicians) write programs using the proposed instruction set? The pragmatists of the last thirty years must be granted that their instruction sets (however inelegant) have been useful to write programs. Human factors suggest that the functions chosen need to have intuitively simple definitions and need to possess "nice" algebraic properties, such as associativity and commutativity.

(2) What are the minimal storage and time requirements needed to evaluate on machine M a given set of functions over E(k)? From the point of view of machine design the optimal set of functions is not a minimal set, but a set that minimizes time and storage requirements over some subset of functions which are "interesting" in one or more applications areas.

Thus, the multiple-valued logician is faced with some new questions. These deal with the formalization of the notion of algorithmic completeness and an analysis of those sets of functions which are complete with constants and admit of a canonical representation of a function f: $E^n(k) \to E(k)$. In addition, which of the maximal closed classes (precomplete classes) [13, 14] contain the functions which applications oriented people find interesting.

Some of these questions are similar to questions arising from the work of Shepherdson and Sturgis [15].

III. Some Theoretical and Experimental Results

A. This author investigated an instruction set with a single operator:

$$Sxyz = \begin{cases} z, & \text{if } x = y; \\ x, & \text{if } x \neq y. \end{cases}$$

For any natural number k, this function is complete with constants over E(k). [4] The choice of operator was inspired by the work of Markov. It is possible (if nerve-racking) to program this way. The writer refrained from publishing the fact that for some one place functions over E(k), the program to evaluate the function takes up at least 19k words of storage.

B. At this Symposium in 1974, this writer showed that for a natural number k, there exists a set of three abelian semigroup operations on E(k) such that the set of functions defined by these operations is complete. [5] Dr. J. C. Muzio has since reduced the number to two and this writer has shown that it cannot be reduced to one.

Specifically, this author showed that if:

$Axy = x + y \pmod k$;

$Mxy = xy \pmod k$; and

E. It appears to be only of theoretical interest that every simple nonabelian group is complete. [9]

IV. From Functions Through Algorithms to Programs

It is not realistic to develop a computer which evaluates functional expressions of arbitrary length. The fundamental notion of a digital computer links it to the notion of an algorithm.

An algorithmic language requires, in addition to the logical operations of the kinds treated earlier in this paper, some instructions which control the flow of the program realizing an algorithm. It is sufficient to provide a "goto" operation which provides transfer of control to a location specified by its argument. This must be accompanied by the labeling of statements. The complete syntax for such a language which the writer has successfully used is contained in Table III.

Flow of control primitives are as important as logical primitives in the design of an optimal language. Their analysis does not appear to be within the scope of classical multiple-valued logic.

The most important study of the effect of the choice of control primitives on the optimality of a language appears to be that of Dr. Louise Jones [10].

V. General and Specific Research Directions

A. Released by IC technology from the old constraints on possible logic and control primitives, how do we develop a set of primitives which are easy to use and which, when applied to known and future applications areas, produce efficient algorithms in terms of size and length of computation?

B. What would be good primitives for list-processing? for string-handling?

C. Specifically, how may the ring $Z(p^n)$ be represented over $GF(p^n)$, both as polynomials and as rational functions? Can ordering be achieved inexpensively as a rational function?

Table I -- Operations for HALL

Mnemonic	Operation	Code (hex)
Arithmetic Unit		
NO	No Operation	0
+B	Binary addition X + Y	1
-B	Binary subtraction X - Y	2
+D	Decimal addition X + Y	3
-D	Decimal subtraction X - Y	4
AN	And X Y	5
OR	Or X Y	6
EX	Exclusive or X Y	7
SL	Shift X left one bit	8
SR	Shift X right one bit	9
Status Unit		
BITO	Set bit to 0	0
BITI	Set bit to 1	1
IBIT	Invert bit	2
DIGO	Set digit to 0	3
DIGI	Set digit to 1	4
IDIG	Invert digit	5
NOOP	No action	7
BZHO	Set bit Z = 0*	8
BZLO	Set bit A = 0	9
DZIO	Set digit Z = 0	A
IBZO	Invert bit Z = 0	B
BZHD	Set bit 0 ≤ Z ≤ 9**	C
BZLD	Set bit 0 ≤ A ≤ 9	D
DZID	Set digit 0 ≤ Z ≤ 9	E

---

* Set bit to 1 if Z = 0.

**Set bit to 1 if Z is a digit, i.e., is a number between 0 and 9.

[2, p. 224]

Table II -- The Instruction Set Proposed for the EDVAC

Instructions consist of <arithmetic instruction> <variation>

They operated on registers I, J, and A.

Arithmetic Instructions

AD	Set A ← I + J.
SB	Set A ← I - J.
ML	Set A ← A + I x J (rounded)
DV	Set A ← $I/\overline{J}$ (rounded)
SQ	Set A ← $\sqrt{I}$ (rounded)
II	Set A ← I
JJ	Set A ← J
SL	If A ≥ 0, set A ← I; if A < 0, set A ← J.
DB	Set A ← binary equivalent of decimal number I.
BD	Set A ← decimal equivalent of binary number I.

Variations

H	Do the operation as described above, holding the result in A.
A	Do the operation as described above, then set J ← I, I ← A, A ← 0.
S	Do the operation as described above, then store the result A into memory location yx and set A ← 0.
F	Do the operation as described above, then store the result into the word immediately following this instruction, set A ← 0, and perform the altered instruction.
N	Do the operation as described above, then store the result into the word immediately following this instruction, set A ← 0, and skip the altered instruction. [11, pp. 250-1]

$$Jxy = \begin{cases} 0, \text{ if } x = 0 \text{ or } y = 0, \text{ not both;} \\ 1, \text{ otherwise;} \end{cases}$$

then {A, M, J} is complete with constants over E(k). Muzio has shown that {A, J} is complete with constants [12].

An attempt was made over the last year to program with the set {A, M, J}. All went well until it was necessary to use some property related to the ordering of E(K) obtained by relativizing the usual ordering of the integers to E(k). Most often the user wants, not E(k) = {0, 1, ..., k-1}, but rather E'(k) = {-[k/2],..., 0,..., [(k-1)/2]}, (where square brackets denote the "greatest integer" function).

C. Order can be handled nicely in E'(k) by introducing the "signum" function:

$$S^*x = \begin{cases} -1, \text{ if } -[k/2] \leq x \leq -1; \\ 0, \text{ if } x = 0; \\ 1, \text{ if } 1 \leq x \leq [(k-1)/2]. \end{cases}$$

In the usual infix notation, we have:

$$Jxy = (S^*((S^*x + S^*y) -1))^2.$$

Hence the set {A, M, S*} is complete with constants. It is not, however, an efficient set with which to program.

Results with the set {A,M,J,S*} have been more hopeful. The author has had undergraduate students produce a selection of routines which represent the facilities available in an average commercially available high level language. The coding involved has been only moderately difficult and is reasonably brief.

As was the case above with the ring Z(k), the problem of inducing an order relation onto $GF(p^n)$ is formidible, that is, the polynomial which corresponds to the order relation:

$$x < y$$

is very long. It appears that the addition of an ordering function might be on the path to optimizing the set of functions.

Since the great body of classical work on divided difference methods is applicable in any field, work with Galois operations has a place to start.

All present computers are either binary or ternary. Hence the Galois fields involved are of characteristic 2 or 3, respectively. This suggests that nothing is to be gained by using subtraction as a primitive. The same is not clear about the inclusion of division as a primitive operation. Programming with rational functions might be far easier than with polynomials alone. There appears to be little research in this area. This investigation into rational functions appears to this writer to be an important research direction.

Table III -- The BNF Grammar for a Rather Primitive Language

```
<program>              ::=  <decl list> | <statement list>
<decl list>            ::=  <decl> | <decl> <decl list>
<decl>                 ::=  <allocation> | <initialization>
<allocation>           ::=  dec <dec identifier> | dec <array designator>
<array designator>     ::=  <dec identifier>     (<number>)
<dec identifier>       ::=  <identifier>
<initialization>       ::=  <allocation> <number>
<statement list>       ::=  <statement> | <statement> <statement list>
<statement>            ::=  <label> : <simple statement>
<label>                ::=  <number>
<simple statement>     ::=  <assignment> | <goto>
<assignment>           ::=  store <arg> at <name>
<goto>                 ::=  goto <arg>
<arg>                  ::=  <expression> | <name> | <number>
<expression>           ::=  <op> <arg> <arg>
<op>                   ::=  A | M | J | S*
<name>                 ::=  <array element> | <identifier>
<array element>        ::=  <identifier>     (<number>)
```

Note: - This grammar is used for an implementation of the language via the XPL system. The redundant productions above are motivated by pragmatic considerations related to the parsing algorithm used by XPL. Further, XPL's Scanning routine produces both <number> and <identifier> as terminal symbols.

Bibliography

1. Joe Weisbecker, "A Simplified Microcomputer Architecture", IEEETC (March, 1974) pp. 41-7.

2. R. H. Evans, L. H. Moffett, and R. E. Merwin, "Design of Assembly Level Language for Horizontal Encoded Microprogrammed Control Unit", Micro-7 Preprints (September, 1974) pp. 217-224.

3. A. J. Nichols, III, "A Microprogramming Framework for Experimental Machine Design", SIGMICRO Newsletter, (July, 1971) pp. 17-21.

4. T. C. Wesselkamper, "A Sole Sufficient Operator", NDJFL, (January, 1975) pp. 86-88.

5. T. C. Wesselkamper, "Some Completeness Results for Abelian Semigroups and Groups", Proceedings of the 1974 International Symposium on Multiple-valued Logic, (May, 1974) pp. 393-400.

6. James T. Ellison, Universal Function Theory and Galois Logic Studies (ARCRL-72-0109) (Bedford, Mass.: Air Force Cambridge Research Laboratories, 1972).

7. B. A. Christensen, J. T. Ellison, R. A. Eggan, Galois Polynomial Generation (PX-7703) (St. Paul: Sperry Rand-Univac, 1972).

8. B. A. Christensen, Notes on Galois Logic Design (PX-10452) (St. Paul: Sperry Rand-Univac, 1973).

9. Heinrich Werner, "Finite Simple Nonabelian Groups are Functionally Complete", Notices AMS, (August 1973) (*73T-A228), p. A-561.

10. Louise H. Jones, "Microinstruction Sequencing for Structured Programming", Micro-7 Preprints (September, 1974) pp. 277-89.

11. Donald E. Knuth, "Von Neumann's First Computer Program", Computing Surveys, (December, 1970), pp. 247-60.

12. J. C. Muzio, "Concerning Completeness and Abelian Semigroups", Zeit. f. Math. Logik u. Grund. d. Math. (to appear).

13. Ivo Rosenberg, "The Number of Maximal Closed Classes in the Set of Functions over a Finite Domain", Journal of Combinatorial Theory 14, no. 1 (January 1973) pp. 1-7.

14. Ivo Rosenberg, "Completeness, Closed Classes and Relations in Multiple-valued Logic", Centre de Recherches Mathematiques, Universite de Montreal (CRM-379), February 1974.

15. J. C. Shepherdson and H. E. Sturgis, "Computability of Recursive Functions", JACM 10, no. 2 (1963) pp. 217-255.

MULTIVALUED LOGIC DESIGN AND

POSTIAN MATRICES

Robert S. Ledley
NBRF
Georgetown University
U.S.A.

H. K. Huang
NBRF
Georgetown University
U.S.A.

It is evident that the utilization of Boolean matrix methods can facilitate the logic design in two-signal levels. [1,2] In this paper we extend the principles of Boolean matrix methods to Postian matrix methods and investigate its application in the multivalued logic design. The matrix methods are based on the manipulation of the designation number of the elementary elements [1] in the system. Therefore in the following the designation numbers, standard basis and Postian Matrices in the Post algebra of order 3 will be introduced first. We shall then discuss the solutions of a Postian Matrix equation in a general case and give examples in the ternary case. The role of Postian Matrices in the three typical problems in logic design will then be investigated.

THE TERNARY CASE

We use the ternary case as an example to introduce the matrix method, the example given in this section should be very easy to extend to a more general case. The ternary case is a Post algebra of order 3, for given v elementary elements, there are 3^v possible input conditions and 3^{3^v} switching functions. In particular, when v=3, the three elementary elements A_1, A_2, A_3 form a standard basis which gives 3^3 possible input conditions, namely:

$\#A_1$ =	012	012	012	012	012	012	012	012	012	*
$\#A_2$ =	000	111	222	000	111	222	000	111	222	
$\#A_3$ =	000	000	000	111	111	111	222	222	222	

The computational techniques in terms of designation numbers can then be directly extended from the two signal levels as soon as the logical operations are defined.

In here, we adopt the formulation indicated for η=3 in the equational axiomization for the disjoint system of Post algebras by Epstein. [3,4] In particular, AND and OR, the greatest lower bound and least upper bound, are represented by $\wedge$ and $\vee$ respectively.

*#A means the designation number for the elementary element A.

In addition, we use the "," to represent the Post's prime operation. [5,6] The logical operation tables become

$\wedge$	0	1	2
0	0	0	0
1	0	1	1
2	0	1	2 ,

$\vee$	0	1	2
0	0	1	2
1	1	1	2
2	2	2	2 ,

and

$0' = 1$

$1' = 2$

$2' = 0$

Now we are ready to discuss the meaning of solutions to a Postian matrix equation. A Postian matrix in the ternary case is a matrix whose elements can take on three values, say 0, 1, and 2. In general, the Postian matrix equation

$$(a_{ik}) \otimes (x_{kj}) = (b_{ij})$$

means

$$b_{ij} = \bigvee_k (a_{ik} \wedge x_{kj})$$

Using the analogy of the solution to the two-valued matrix equations,[1] we shall find an (x'_{kj}) solution such that, if (x_{kj}) is also a solution, then $(x_{kj}) \rightarrow (x'_{kj})$.* Here we use the $\rightarrow$ in the sense $0 \rightarrow 1 \rightarrow 2$, $0 \rightarrow 0$, $1 \rightarrow 1$, $2 \rightarrow 2$. Also $(p_{kj}) \rightarrow (q_{kj})$ if $p_{kj} \rightarrow q_{kj}$ for each k and j. In order to determine our (x'_{kj}), let us first observe that for a particular (a_{ik}) and (b_{ij}) the greatest value that x_{kj} can take so that $a_{ik} \wedge x_{kj} \rightarrow b_{ij}$ is as follows: (where "greatest" is in terms of our $\rightarrow$ relation):

a_{ik}	012	012	012
b_{ij}	000	111	222
x_{kj}	200	221	222

For example, if $a_{ik} = 0$ and $b_{ij} = 0$, then $0 \wedge 2 = 0 \rightarrow 0$ so that x_{kj} can be 0, 1, 2; the remaining eight cases are treated similarly. In terms of our three-valued operations we may write (omitting the subscripts) $x=(aVa')'\wedge(bVb'')'\wedge a'Vb$.

THE c PRODUCT

From now on, we shall consider the general case of Post algebra of order $n+1$.[5,6,7] Let us first define the noncommutative c product as follows:

c	e_0	e_1	e_2	e_3 . . .	e_n
e_0	e_n	e_n	e_n	e_n . . .	e_n
e_1	e_0	e_n	e_n	e_n . . .	e_n
e_2	e_0	e_1	e_n	e_n . . .	e_n
e_3	e_0	e_1	e_2	e_n . . .	e_n
...					
e_n	e_0	e_1	e_2	e_3 . . .	e_n

*Not to confuse the "$\rightarrow$" with the intuitionist implication $\rightarrow$ defined by Gödel and Heyting.[3] Their intuitionist implication is our "c" product to be defined later.

Where e_0, e_1 . . ., e_n are the constants defined by Epstein. [3] In the ternary case, $e_0=0$, $e_1=1$, $e_2=2$. Based on this definition, consider the c matrix operation

$$(a_{ki}) \textcircled{C} (b_{ij}) = (c_{kj})$$

defined by $$c_{kj} = \bigwedge_i (a_{ki} \; c \; b_{ij})$$

Then $$(x'_{kj}) = (a_{ik})^T \textcircled{C} (b_{ij})$$

is the desired solution. If a solution to our original matrix equation exists, then

$$(a_{ik}) \otimes [(a_{ik})^T \textcircled{C} (b_{ij})] = (b_{ij})$$

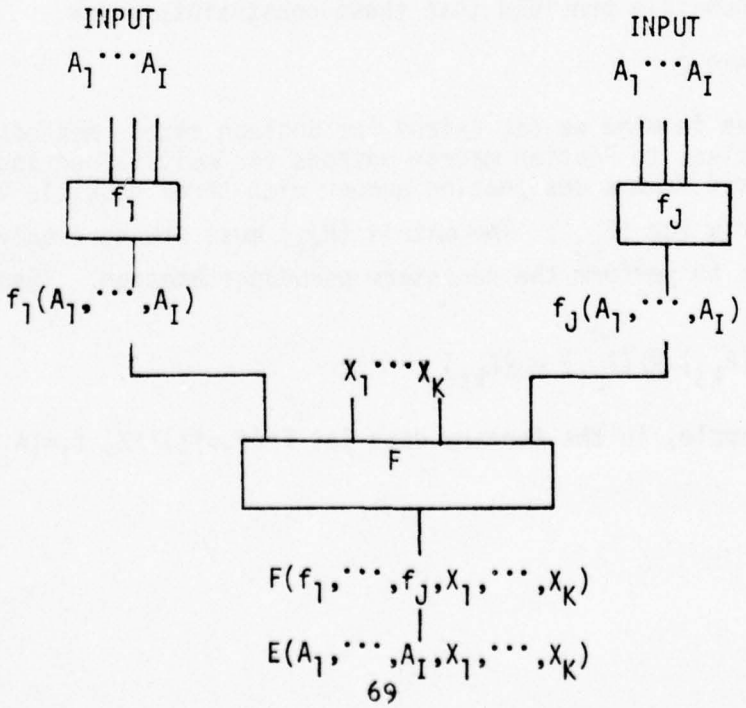
Otherwise $$(a_{ik}) \otimes [(a_{ik})^T \textcircled{C} (b_{ij})] \rightarrow (b_{ij})$$

It is interesting to know at this point that the c product just defined is identical to the intuitionist implication of Gödel and Heyting [3] modified by Epstein. In the ternary case, this implication is defined as $u \rightarrow v = (e_1 \wedge [C_2(u) \wedge C_1(v)]) \vee [C_0(u) \vee (C_1(u) \wedge C_1(v)) \vee C_2(v)]$, where C_i are the operators defined by Epstein. Hence in terms of the implication, the matrix element c_{kj} can be represented by

$$c_{kj} = \bigwedge_i (a_{ki} \rightarrow b_{ij})$$

THE THREE TYPICAL PROBLEMS IN LOGIC DESIGN

Consider the following figure which is a block diagram of a logic design that has been separated into two subunits f_1, . . ., f_j and F,



The three typical problems for this logic design are as follows [1]:

1. The function F $(f_1,\ldots f_j, X_1,\ldots,X_K)$ is given explicitly, and the functions $f_1\ (A_1,\ldots,A_I),\ldots,f_J(A_1,\ldots,A_I)$ are each given explicitly; the problem is to find the unknown function $E(A_1,\ldots,A_I, X_1,\ldots,X_K)$.

2. The function E $(A_1,\ldots,A_I, X_1,\ldots,X_K)$ is given explicitly, and the functions $f_1\ (A_1,\ldots,A_I),\ldots,f_J(A_1,\ldots,A_I)$ are each given explicitly; the problem is to find the unknown function F $(f_1,\ldots,f_J, X_1,\ldots,X_K)$ such that E = F.

3. Both the switching functions $E(A_1,\ldots,A_I,X_1,\ldots,X_K)$ and $F(f_1,\ldots,f_J,X_1,\ldots,X_K)$ are given explicitly; the problem is to find unknown functions $f_1(A_1,\ldots,A_I),\ldots,f_J(A_1,\ldots,A_I)$ such that E = F.

The first problem presents no difficulty, for in order to find E we merely substitute in F the explicit expression for each f_j of the A_i. The E so determined is always such that F = E.

The second and third problems are more difficult to solve, and systematic methods for their solution involving the Postian Matrix methods will be given in the next sections. A solution can always be found for the second problem; this is not so for the third problem. It may happen in certain instances that no solutions to the third problem exist. In such an event solutions exist only if certain constraints hold between $A_1, A_2,\ldots,A_I$. These constraints may then be obtained, as well as the solutions which are possible provided that these constraints occur.

PROBLEMS OF TYPES 1 AND 2

With the above in mind we can extend the Boolean matrix methods for antecedence and consequence solutions to Postian matrix methods for multivalued logic. The matrix (E_{ki}) [1] is now formed from a designation number with three possible values for each position, and similarly for (F_{kj}). The matrix (R_{ji}) must now have only a single 2 in each column, in order to perform the necessary pseudopermutation. Then the fundamental equation

$$(F_{kj}) \otimes (R_{ji}) = (E_{ki})$$

still holds. For example, in the ternary case let $F=(f_1 \vee f_2')' \wedge X$, $f_1=(A_1 \wedge A_2')'$ and $f_2=(A_1 \vee A_2')'$.

Then

$$\begin{array}{lll} \#f_1 = 012\ 012\ 012 & 012\ 012\ 012 & 012\ 012\ 012 \\ \#f_2 = 000\ 111\ 222 & 000\ 111\ 222 & 000\ 111\ 222 \\ \underline{\#X = 000\ 000\ 000} & \underline{111\ 111\ 111} & \underline{222\ 222\ 222} \\ \#F = 000\ 000\ 000 & 110\ 000\ 110 & 220\ 000\ 120 \end{array}$$

$$\begin{array}{lll} \#A_1 = 012 & 012 & 012 \\ \#A_2 = 000 & 111 & 222 \\ \hline \#f_1 = 122 & 120 & 111 \\ \#f_2 = 220 & 000 & 120 \end{array}$$

Hence

$$(F_{kj}) = \begin{bmatrix} 000 & 000 & 000 \\ 110 & 000 & 110 \\ 220 & 000 & 120 \end{bmatrix} \quad \text{and} \quad (R_{ji}) = \begin{bmatrix} 000 & 002 & 000 \\ 000 & 200 & 002 \\ 002 & 020 & 000 \\ 000 & 000 & 000 \\ 000 & 000 & 200 \\ 000 & 000 & 000 \\ 000 & 000 & 000 \\ 200 & 000 & 020 \\ 020 & 000 & 000 \end{bmatrix}$$

Then

$$(F_{kj}) \textcircled{x} (R_{ji}) = \begin{bmatrix} 000 & 000 & 000 \\ 100 & 101 & 011 \\ 200 & 202 & 022 \end{bmatrix} = (E_{ki})$$

or

$$\#E = 000\ \ 000\ \ 000\ \ 100\ \ 101\ \ 011\ \ 200\ \ 202\ \ 022$$

and

$$E=\{A_1VA_1')'\wedge(A_2VA_2')'V(A_1\wedge A_1')''\wedge(A_2VA_2'')'V(A_1'\wedge A_1'')''\wedge(A_2'VA_2')'\}\wedge X.$$

Thus it is clear how to solve problems of type 1, that is, substitution problems. For type 2 problems we solve the fundamental equation by finding

$$(F_{jk}) = (R_{ji}) \textcircled{c} (E_{ik})$$

and then substitute into

$$(F_{kj}) \textcircled{x} (R_{ji}) = (E'_{ki})$$

to determine whether or not $(E'_{ki}) = (E_{ki})$, that is, whether or not a solution to the logic design problem actually exists.

Suppose that in the ternary case for the same f_1, f_2 and E as before; then, to find F, we compute

$$\begin{bmatrix} 000 & 002 & 000 \\ 000 & 200 & 002 \\ 002 & 020 & 000 \\ \\ 000 & 000 & 000 \\ 000 & 000 & 200 \\ 000 & 000 & 000 \\ \\ 000 & 000 & 000 \\ 200 & 000 & 020 \\ 020 & 000 & 000 \end{bmatrix} \textcircled{c} \begin{bmatrix} 012 \\ 000 \\ 000 \\ \\ 012 \\ 000 \\ 012 \\ \\ 000 \\ 012 \\ 012 \end{bmatrix} = \begin{bmatrix} 012 \\ 012 \\ 000 \\ \\ 222 \\ 000 \\ 222 \\ \\ 222 \\ 012 \\ 000 \end{bmatrix} = (F_{jk})$$

Since f_1 and f_2 are constrained, we find F with respect to the constrained basis,

$\#f_1$ =	012	0	1	2	12	012	0	1	2	12	012	0	1	2	12
$\#f_2$ =	000	0	1	2	22	000	0	1	2	22	000	0	1	2	22
#X =	000	0	0	0	00	111	1	1	1	11	222	2	2	2	22
#F =	000	0			00	110	0			10	220	0			20

which is now the same as the F used above.

There are two other equivalent forms to the fundamental equation which follow directly, since (R_{ji}) is a pseudopermutation:

$$(F'_{kj}) \textcircled{x} (R_{ji}) = (E'_{ki}) \text{ and } (F''_{kj}) \textcircled{x} (R_{ji}) = (E''_{ki})$$

These give rise to three types of logical problems: antecedence, intercedence, and consequence problems as in the two-valued logic.

PROBLEMS OF TYPE 3

The multivalued generalization of the θ matrix product is particularly useful for problems of Type 3. The θ product in a Post algebra of order $n+1$ is defined as

$$\theta(e_i, e_j) = e_n \qquad i=j$$
$$= 0 \qquad i \neq j$$

The θ matrix product is then defined by

$$(a_{ki}) \textcircled{\theta} (b_{ij}) = (c_{kj})$$

where $$c_{kj} = \bigwedge_i (a_{ki} \theta b_{ij})$$

With this in mind we have from the fundamental equation that

$$(R_{ji}) = (F_{kj})^T \textcircled{\theta} (E_{ki})$$

from which the solutions can be found that have a single e_η in each column. For example, in the ternary case, with the same F and E as given before, then

$$(R_{ji}) = \begin{bmatrix} 012 \\ 012 \\ 000 \\ \\ 000 \\ 000 \\ 000 \\ \\ 011 \\ 012 \\ 000 \end{bmatrix} \textcircled{\theta} \begin{bmatrix} 000 & 000 & 000 \\ 100 & 101 & 011 \\ 200 & 202 & 022 \end{bmatrix} = \begin{bmatrix} 200 & 202 & 022 \\ 200 & 202 & 022 \\ 022 & 020 & 200 \\ \\ 022 & 020 & 200 \\ 022 & 020 & 200 \\ 022 & 020 & 200 \\ \\ 000 & 000 & 000 \\ 200 & 202 & 022 \\ 022 & 020 & 200 \end{bmatrix}$$

The $3X5X5X3X5X3X5X3X3 = 151{,}875$ sets of solutions include the set $f_1=(A_1 \wedge A_2')'$, $f_2=(A_1 \vee A_2')'$. The θ product just given can be related to the strict implication $\Rightarrow$ of Epstein. [3] In the ternary case, this implication is defined as

$$u \Rightarrow v = C_o(u) \vee (C_1(u) \wedge C_1(v)) \vee C_2(v)$$

where C_i are the operators as before and it can be extended to a more general case. With this definition in mind, we have

$$\theta(u,v)=(u \Rightarrow v) \wedge (v \Rightarrow u)$$

Therefore the θ matrix product can be represented by

$$c_{kj}=\bigwedge_i((a_{ki} \Rightarrow b_{ij}) \wedge (b_{ij} \Rightarrow a_{ki}))$$

Moreover, since

$$C_i(x) = (x \Rightarrow e_i) \wedge (e_i \Rightarrow x),$$

we can relate the θ product to the C_i operators by the following equation

$$C_i(x) = \theta(x, e_i).$$

It is also interesting to note the relation between the θ and the c operations at this point, namely,

$$(a_{ki}) \textcircled{\theta} (b_{ij}) = \{(a_{ki}) \textcircled{c} (b_{ij})\} \wedge \{(a'_{ki}) \textcircled{c} (b'_{ij})\} \wedge \{(a''_{ki}) \textcircled{c} (b''_{ij})\}$$

ALL SOLUTIONS TO POSTIAN MATRIX EQUATIONS IN A MULTIVALUED LOGIC

Let $\textcircled{M}$ be any matrix-type logical operation [that is, for $(c_{pr}) = (a_{pq}) \textcircled{M} (b_{qr})$, each c_{pr} is some logical function of corresponding (qth) elements of the pth row of (a_{pq}) and the rth column of (b_{qr})]. Then if

$$(a_{ik}) \textcircled{M} (x_{kj}) = (b_{ij})$$

all matrix solutions, if any exist, if any exist, can be found from

$$(S_{mj}) = \{(a_{ik}) \textcircled{M} (b[X_1, \ldots, X_K])\}^T \textcircled{\theta} (b_{ij})$$

by the usual R-matrix type of procedure, [1] generalized as required. The operations $\textcircled{\theta}$ and $\textcircled{c}$ can be considered as specialized examples of the general matrix operation $\textcircled{M}$.

CONCLUSION

We have investigated the utilization of the Postian matrix methods in multivalued logic design by extending the c and θ products, the c and θ matrix products, the E, F, R, S matrices from two-valued levels to multivalued levels. The c and θ products so defined are related to the Gödel and Heyting's intuitionist implication and Epstein's strict implication respectively and hence are also functions of Epstein's C_i operators.

R E F E R E N C E S

[1] R.S. Ledley, Digital Computer and Control Engineering, McGraw-Hill, 1960.

[2] S. Rudeanu, Boolean Functions and Equations, pp. 371-382, North-Holland, 1974.

[3] G. Epstein, G. Frieder and D.C. Rine, "The Development of Multiple-valued logic as related to Computer Science", Computer, vol. 7 no. 9, pp. 20-32, Sept. 1974.

[4] G. Epstein, "An Equational Axiomatization for the Disjoint System of Post Algebras, IEEE Tran. Computers, vol. C-22, no. 4, pp. 422-423, 1973.

[5] P.C. Rosenbloom, "Post Algebras I. Postulates and General Theory", Am. J. Math., vol. 64, pp. 167-188, 1942.

[6] G. Epstein, "The Lattice Theory of Post Algebras", Tran. Am. Math. Soc., vol. 95, pp. 300-317, 1960.

[7] E.L. Post, "Introduction to a General Theory of Elementary Propositions, Am. J. Math., vol. 43, pp. 163-185, 1921.

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 $D = \{0, 1, 2, \dots, \not{d}\}$;
$\mathbf{E}(d_1, d_2, \dots d_n)$ or $\mathbf{E}$	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 $\phi - V$;
$V_1 V_2$ 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):

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

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 $0 \leq \tau_i \leq 1$, $i = 1 , 2 , \ldots , \ell$, 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

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

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

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

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 $A = \langle \text{a-list} \rangle$. 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 $\mathcal{L} \subseteq 2^{\Omega}$ is called universe of complexes, if it satisfies:

(i) coverability criterion:

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

(ii) separability criterion:

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

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 $E_2 = \Omega \backslash E_2$.

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:

$$\mathbf{R}_j = \bigcap_{i \in I} \{x_i = \alpha_i\} \tag{4}$$

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 $\mathbf{E}$.

Definition 2. The cartesian (interval) root, $\sqrt{E}$ ($\sqrt[\cdot]{E}$) , 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:

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

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

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

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:

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

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 $G(E_1|E_2) = \phi$, and if $E_1 \cap E_2 \neq \phi$ then $CG(E_1|E_2) \neq \phi$. 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, $G(e|E_2)$, $e \in E^r$, $E^r \subseteq E_1$, $E_1 \cap E_2 = \phi$. 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:

$$c(M^q) - c(M) \geq \Delta \qquad (12)$$

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 features.

5. Algorithm A^q. The basic idea of the algorithm A^q for the synthesis of a cover $CV(E_1|E_2)$ 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 $A = \langle a\text{-list}, \tau\text{-list} \rangle$, 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 $G^{\cup}(e|E_2)$, then the computed Δ ceases to be a true estimate. (12)

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 $S_k = \{e\} \vdash \{\overline{e}_k\}$ for each $e_k \in E$.

Suppose $e_k = \{x_1 = a_1\}\{x_2 = b_2\} \dots \{x_n = a_n\}$ (14)

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

properties [6]:

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

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

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:

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

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 $d-1$, ..., 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

$$\{{}^{j}f : \mathbf{E} \rightarrow {}^{j}D\}_{j=1,2,\ldots,m} \tag{23}$$

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

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

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 ${}^jD$, $j = 1, 2, \ldots, m$, can be any finite sets and there are no restrictions on F^{jk}.

If D_i, $i = 1, 2, \ldots, n$ and ${}^jD$, $j = 1, 2, \ldots, m$ are all $\{0, 1\}$, then f becomes a multiple output binary switching function. When ${}^jD$ are not binary, an important interpretation of f can be as a 'multivalued non-unique' decision function. Elements of ${}^jD$ can be interpreted as 'degree of truth' or 'confidence degree' that an event $e \in F^{jk}$, $k \in {}^jD$, 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 = <a-list, τ=list>, where elements of a-list are attributes which characterize whole set $\{V({}^jf)\}$ of VL_1 expressions of ${}^jf$.

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

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

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

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.

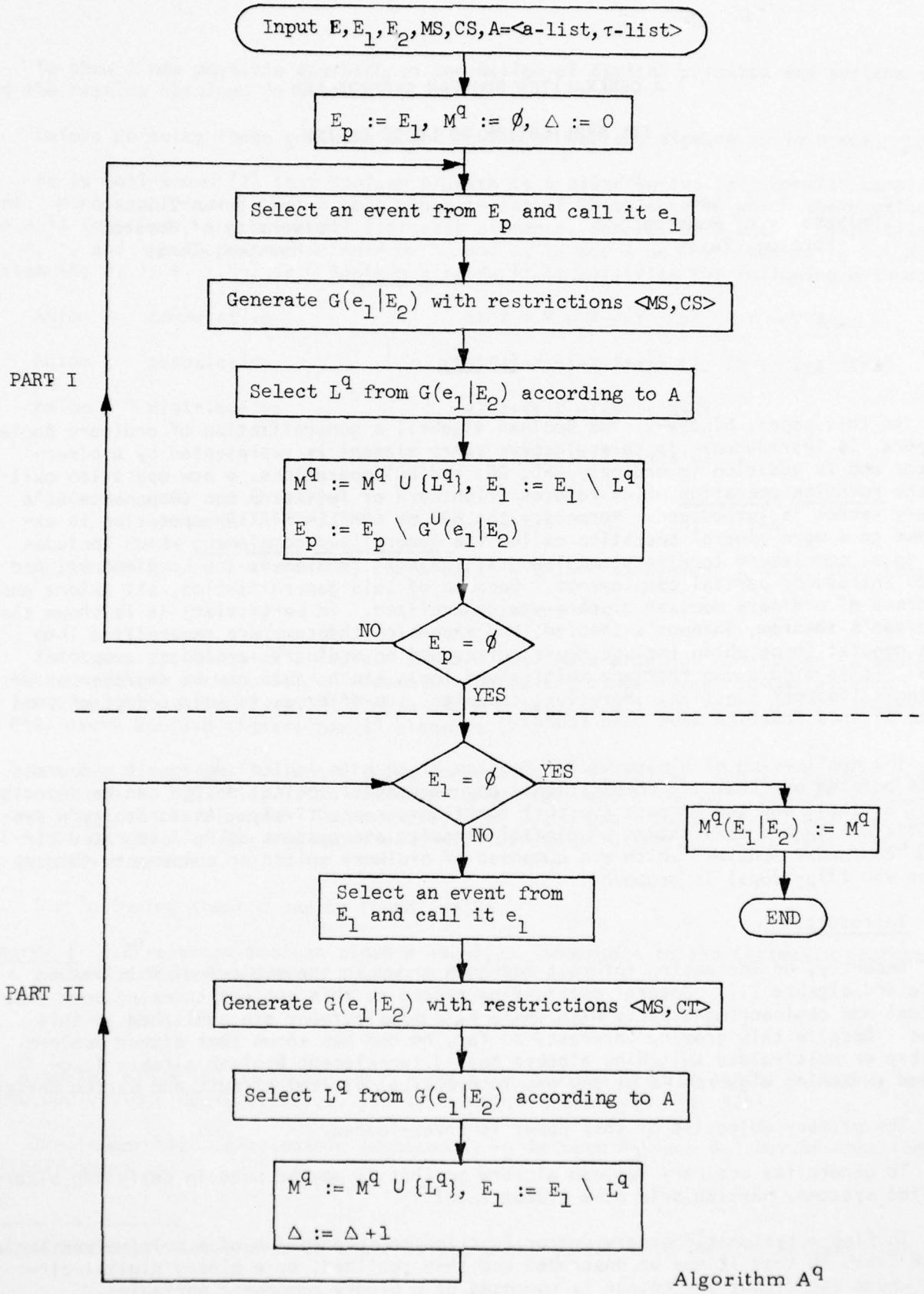
9. Michalski, R. S., "Synthesis of Minimal Forms and Recognition of Symmetry of Switching Functions," Proceedings of the Institute of Automatic Control, No. 92, Polish Academy of Sciences, Warsaw 1971 (in Polish).

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

A GENERALIZED BOOLEAN ALGEBRA AND ITS APPLICATION TO LOGIC DESIGN

S. C. Lee
University of Houston
Houston, Texas

Y. Keren-Zvi
University of Houston
Houston, Texas

Summary

In this paper, binary-vector Boolean algebra, a generalization of ordinary Boolean algebra, is introduced. In this algebra, every element is represented by a binary vector and in addition to ordinary AND, OR, and NOT operations, a new operation called the rotation operation which rotates (rightward or leftward) the components of a binary vector is introduced. Moreover, the NOT or COMPLEMENTATION operation is extended to a more general operation called the generalized complement which includes the total complement (ordinary complement), the null complement (no complement), and newly introduced partial complements. Because of this generalization, all axioms and theorems of ordinary Boolean algebra are generalized. In particular, it is shown that DeMorgan's theorem, Shannon's theorem, and expansion theorems are generalized into more general forms which include their corresponding ordinary version as a special case. It is also shown that any multivalued logic truth table can be represented by a single (vector) function. Moreover, canonical sum-of-products and product-of-sums forms of this function have been found.

The application of binary-vector Boolean algebra to logical design is discussed. It is pointed out that any combinational and sequential logical design can be described by a single and two compact (vector) functions, respectively. As a result, a systematic design and realization of digital circuits and systems using integrated circuit "component modules" which are composed of ordinary switching components (binary gates and flip-flops) is proposed.

1. Introduction

Recently, an increasing interest has been shown in the subject of multivalued logic and algebra [1]. Several outstanding papers on this subject covering both theoretical and engineering aspects with up-to-date bibliography are published in this issue. Despite this growing interest, so far, no one has shown that either Boolean algebra or multivalued switching algebra beyond two-element Boolean algebra (two-valued switching algebra) is of any use in practical digital circuit and system design.

The primary objective of this paper is three-folded.

1. To generalize ordinary Boolean algebra so that it may be used in designing binary digital systems, particularly at a system level.

2. To find a systematic binary-vector function representation of a multivalued logic truth table so that it may be described and then realized, by a binary digital circuit whose each input and output is composed of n binary component variables.

3. To show a new possible approach to the design of digital circuits and systems using the results obtained in 1 and 2.

Before pursuing these goals, a brief review of Boolean algebra is in order.

It is well-known [2] that Boolean algebra is a distributive lattice with complement. More specifically, let B be a nonempty set of 2^n objects in which there exists the null (smallest) and universal (largest) elements, denoted by 0 and 1, respectively, +, ·, and - be operators defined on B, and X, Y, and Z be any elements in B. A system (B; 0, 1; +, ·, -) is a Boolean algebra if it satisfies the following axioms:

Axiom 1	commutative	a) $X + Y = Y + X$ b) $X \cdot Y = Y \cdot X$
Axiom 2	associative	a) $(X+Y)+Z=X+(Y+Z)$ b) $(X \cdot Y) \cdot Z=X \cdot (Y \cdot Z)$
Axiom 3	distributive	a) $X+(Y \cdot Z)=(X+Y) \cdot (X+Z)$
		b) $X \cdot (Y+Z)=(X \cdot Y)+(X \cdot Z)$
Axiom 4	null and universal elements	a) $X+0=X$ b) $X \cdot 1=X$
Axiom 5	For each element X, there exist a unique complement $\overline{X}$ of X, such that	
	a) $X+\overline{X}=1$	b) $X \cdot \overline{X}=0$

Since every element of a Boolean algebra must have a unique complement, a Boolean algebra must have even number of elements. Moreover, since every Boolean algebra is isomorphic* to a power set P(A) of a set $A = \{a_1,\ldots,a_n\}$ and there are 2^n elements in P(A) every Boolean algebra has 2^n elements [2].

Let B_{2n} denote the Boolean algebra containing 2^n elements. The B_2, B_4, B_8, and B_{16} are shown in Fig. 1(a). The elements immediately above the null element are called the atoms of the algebra. For example, I of B_2, a and b of B_4, a, b, and c of B_8, and a, b, c, and d of B_{16} are atoms of their respective algebra. It is seen that B_2^n has exactly n atoms.

The following theorem may be found in [3].

Theorem 1 A 2^n-element Boolean algebra (B_{2^n}) is isomorphic to the Cartesian product of n 2-element Boolean algebras $B_2=\{0,1\}$; namely,

$$B_{2^n} = B_2 \times B_2 \times \ldots \times B_2 = B_2^n$$

For example, $B_2^2 = \{00,01,10,11\}$, $B_2^3 = \{000,001,010,100,011,101,110,111\}$, and $B_2^4 = \{0000,0001,0010,0100,1000,0111,\ldots,1111\}$, which are shown in Fig. 1(b).

The isomorphism relationship (denoted by $\sim$) between B_{2^n} and B_2^n may be described by their atoms as:

*Two algebras are isomorphic if there exists a one-to-one onto mapping between them.

$B_2 \sim B_2^1$	$B_4 \sim B_2^2$	$B_8 \sim B_2^3$	$B_{16} \sim B_2^4$
I=1	a=01 b=10	a=001 b=010 c=100	a=0001 b=0010 c=0100 d=1000

which are shown in Fig. 1. If decimal numbers are used to represent the elements of B_2^n, n=1,2,3 and 4, they are shown in Fig. 1(c).

II. The Generalized Complement and the Rotation Operation

In ordinary Boolean algebra, one type of complement is defined, namely, the total complement. For example, in B_2^3, the (total) complement of the elements are

X	$\overline{X}$ (total complement)
000	111
001	110
010	101
100	011
011	100
101	010
110	001
111	000

The total complement operation $\overline{X}$ is the one that complements every component of X.

Definition 1 In B_2^n, the generalized complement X^P, $P \varepsilon B_2^n$, of a variable X is defined to be the element obtained from complementing the components of X according to the value of the corresponding component of P; x_i is complemented and uncomplemented if p_i is 0 and 1, respectively, where x_i and p_i, designate the ith component of X and P.

For example, the generalized complement of the elements of B_2^3, for P=010 are

X	X^P
000	101
001	100
010	111
100	001
011	110
101	000
110	011
111	010

It is important to point out that the generalized complement includes the uncomplement (true form) and the total complement as special cases.

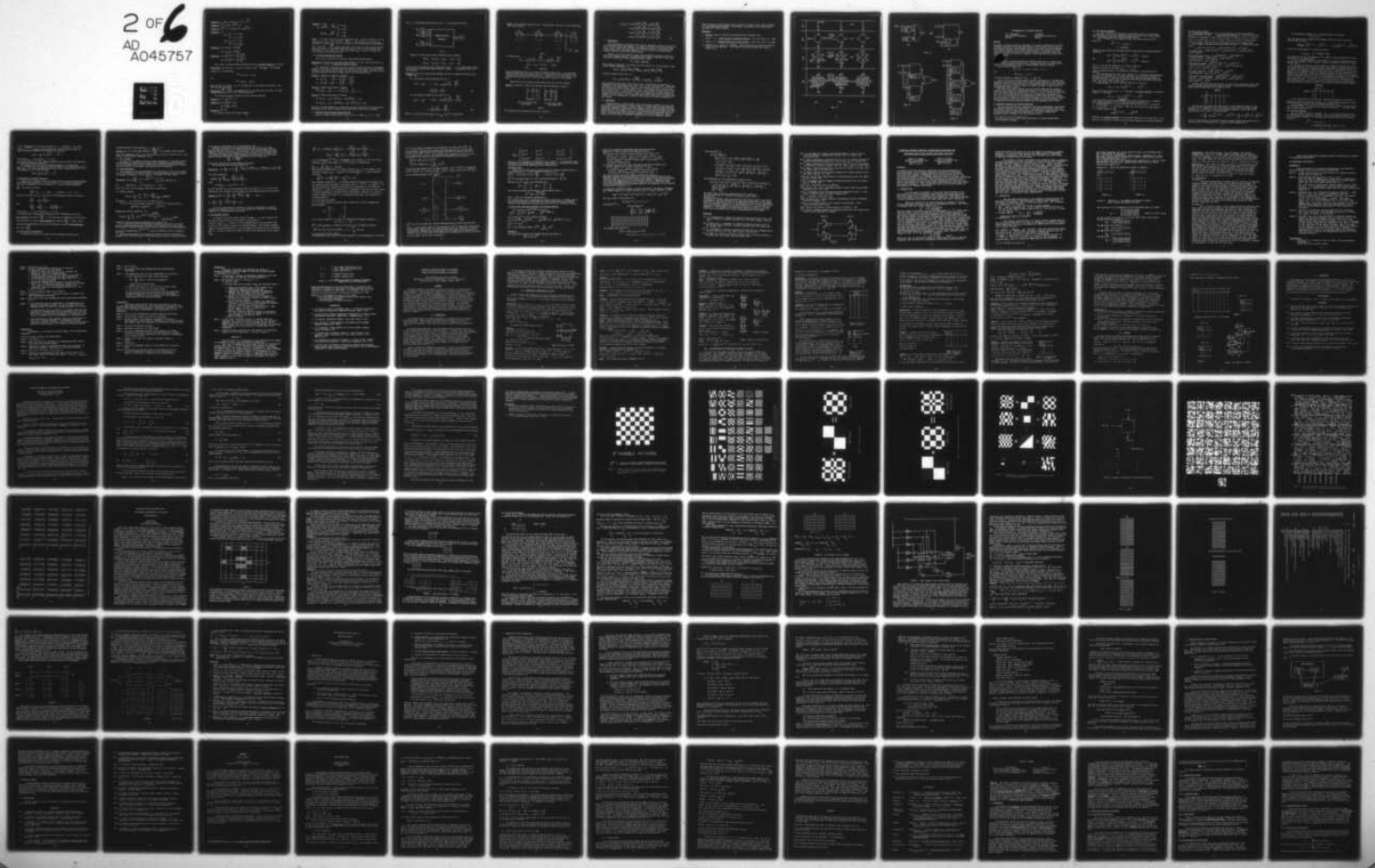
The following are several basic properties of the generalized complement.

Property 1 $X^0=\overline{X}$ and $X^I=X$, where $0=(0,\ldots,0)$ and $I=(1,\ldots,1)$.

Property 2 (a) $\overline{P}^P=I$.
(b) $\overline{P}^P=P^{\overline{P}}=\overline{P}^{\overline{P}}=0$.

AD-A045 757 INDIANA UNIV BLOOMINGTON DEPT OF COMPUTER SCIENCE F/G 9/2
PROCEEDINGS OF THE 1975 INTERNATIONAL SYMPOSIUM ON MULTIPLE-VAL--ETC(U)
MAY 75 G EPSTEIN N00014-75-C-0449
UNCLASSIFIED MVL-75-001 NL

2 OF 6
AD A045757



Property 3 $P_1 = \overline{P}_2$ if and only if $X^{P_1} = X^{\overline{P_2}}$.

Property 4 $X^{PP} = (X^P)^P = X^{(PP)} = X$.

Property 5 $X^{P^{2n+1}} = X^P$, where $X^{P^m} = ((X^{\overbrace{P)\ldots)^P}^{m}}$

Property 6 (a)

$$P^{P^m} = \begin{cases} I & \text{if } m \text{ is odd} \\ P & \text{if } m \text{ is even} \end{cases}$$

$$P^{\overline{P}^m} = \begin{cases} 0 & \text{if } m \text{ is odd} \\ \overline{P} & \text{if } m \text{ is even} \end{cases}$$

Property 7 (a) $(X_1+X_2)^P = (\overline{X}_1 \cdot \overline{X}_2)^{\overline{P}}$

(b) $(X_1 \cdot X_2)^P = (\overline{X}_1+\overline{X}_2)^{\overline{P}}$

Property 8 (a) $(X_1^{P_1}+X^{P_2})^{P_3} = (X_1^{\overline{P}_1} \cdot X_2^{\overline{P}_2})^{\overline{P}_3}$

(b) $(X_1^{P_1} \cdot X_2^{P_2})^{P_3} = (X_1^{\overline{P}_1}+X_2^{\overline{P}_2})^{\overline{P}_3}$

Now we shall introduce another operation called the rotation operation as follows.

Definition 2 Let $X=(x_1,x_2,\ldots,x_n)$ be a variable in B_2^n. The right and the left rotation of X are defined as

$$\vec{X} = (x_n,x_1,x_2,\ldots,x_{n-1})$$

and

$$\overleftarrow{X} = (x_2,x_3,\ldots,x_n,x_1)$$

When the arrow is omitted, i.e., $\tilde{X}$, it means that it can either be the right or the left rotation operation.

Definition 3 $\overbrace{\tilde{X}}^{(m)} = (\overbrace{(\tilde{X})}^{\text{i}})$. The operations $\sim$ in this definition are either all right rotation operations or all left rotation operations.

The following are basic properties of the rotation operation.

Property 9 If $X \in B_2^n$, $\overbrace{\tilde{X}}^{(n)} = X$.

Property 10 (a) $(\widetilde{X_1 \cdot X_2}) = \tilde{X}_1 \cdot \tilde{X}_2$.

(b) $(\widetilde{X_1+X_2}) = \tilde{X}_1+\tilde{X}_2$.

Property 11 $\widetilde{X^P} = \tilde{X}^{\tilde{P}}$

It is easy to show the following theorem.

Theorem 2 In B_2^n,

$$\text{(a)}\quad (X^P)(\tilde{X}^P)\ldots(\tilde{X}^{(n-1)P}) = \begin{cases} I & \text{if } X=P \\ 0 & \text{if } X\neq P. \end{cases}$$

$$X^{\overline{P}}+(X^{\overline{P}})+\ldots+(\tilde{X}^{(n-1)\overline{P}}) = \begin{cases} 0 & \text{if } X=P \\ I & \text{if } X\neq P \end{cases}$$

Proof: It is easily seen that if $X=P$, by Property 2(a) the above expression is true. The reason for that if $X\neq P$, $(X^P)(\tilde{X}^P)\ldots(\tilde{X}^{(n-1)P})=0$, is that there will be at least one component of X which is 0. This 0 is rotated in X by the operation $\sim$. The presence of the n terms $(X^P),\ldots,(\tilde{X}^{(n-1)P})$, ensures that there will be one 0 in every position of the vector X, which is in (at least) one of the n terms. This, in turn, ensures that the product of the n terms is 0. Hence the theorem 2(a). Similarly, theorem 2(b) can be proven by using Property 2(b).

III. The Generalized Boolean Algebra

Now we are in the position to define a generalized Boolean algebra.

Definition 4 The B_2^n with the generalized complement and the rotation operation defined on it is called the generalized Boolean algebra.

For convenience, from now on, we shall use B_2^n to denote the generalized Boolean algebra. Since B_2^n is a generalization of ordinary Boolean algebra, all the axioms, properties, and theorems of the latter are satisfied by the former. Moreover, all the important theorems in ordinary Boolean algebra can now be generalized.

Theorem 3 (Generalized DeMorgan's Theorem)

$$\text{(a)}\quad (X_1^{P_1} X_2^{P_2} \ldots X_m^{P_m})^P = (X_1^{\overline{P}_1}+X_2^{\overline{P}_2}+\ldots+X_m^{\overline{P}_m})^{\overline{P}}$$

$$\text{(b)}\quad (X_1^{P_1}+X_2^{P_2}+\ldots+X_m^{P_m})^P = (X_1^{\overline{P}_1} X_2^{\overline{P}_2} \ldots X_m^{\overline{P}_m})^{\overline{P}}$$

Theorem 4 (Generalized Shannon's Theorem)

$$f^P(X_1^{P_1},X_2^{P_2},\ldots,X_m^{P_m};\ +,\ \cdot) = f^{\overline{P}}(X_1^{\overline{P}_1},X_2^{\overline{P}_2},\ldots,X_m^{\overline{P}_m};\ \cdot,\ +)$$

Theorem 5 (Generalized Expansion Theorem)

$$\text{(a)}\quad f(X_1,X_2,\ldots,X_m) = X_1^P f(P,X_2,\ldots,X_m)+X_1^{\overline{P}} f(\overline{P},X_2,\ldots,X_m)$$

$$\text{(b)}\quad f(X_1,X_2,\ldots,X_m) = (X_1^P+f(\overline{P},X_2,\ldots,X_m))\cdot(X_1^{\overline{P}}+f(P,X_2,\ldots,X_m))$$

The proofs of these theorems are evident from the above properties of the generalized complement and the rotation operation described above, and may thus be omitted.

IV. Canonical Forms of Vector Boolean Function

Consider a general combinational network of Fig. A, where x_{ij}, i=1,...,m and

$j=1,\ldots,n$, are binary variables and f_k, $k=1,\ldots,n$, are binary functions.

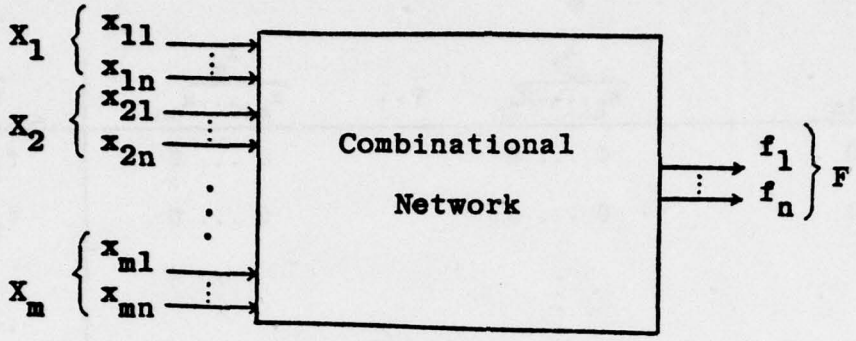


Fig. A

Normally, we would need n output functions of m×n variables,

$$f_1(x_{11},\ldots,x_{1n},x_{21},\ldots,x_{2n},\ \ldots,\ x_{m1},\ldots,x_{mn})$$
$$\vdots$$
$$f_n(x_{11},\ldots,x_{1n},x_{21},\ldots,x_{2n},\ \ldots,\ x_{m1},\ldots,x_{mn})$$

In this section, we show that this network may be described by a single vector output function of m n-tuple binary-vector variables. Moreover, two canonical forms of it are presented.

Theorem 6 Let F be a n-tuple vector Boolean function of m Boolean variables $X_1, X_2, \ldots, X_m$. Then

(a) the canonical sum-of-products form of F is

$$F(X_1,\ldots,X_m) = \sum F(P_1,\ldots,P_m)[X_1^{P_1}(\widetilde{X_1^{P_1}})\ldots(\overbrace{X_1^{P_1}}^{(n-1)})]$$
$$\ldots [X_m^{P_m}(\widetilde{X_m^{P_m}})\ldots(\overbrace{X_m^{P_m}}^{(n-1)})] \qquad (1)$$

(b) the canonical product-of-sums form of F is

$$F(X_1,\ldots,X_m) = \prod F(P_1,\ldots,P_m)+[X_1^{\overline{P}_1}+(\widetilde{X_1^{\overline{P}_1}})+\ldots+(\overbrace{X_1^{\overline{P}_1}}^{(n-1)})]$$
$$+\ldots+[X_m^{\overline{P}_m}+(\widetilde{X_m^{\overline{P}_m}})+\ldots+(\overbrace{X_m^{\overline{P}_m}}^{(n-1)})] \qquad (2)$$

where $P_1,\ldots,P_m$ take on the values 0,1,2,..., and 2^n in binary form.

Proof: First consider the proof of (a). The function F (see Fig. 1) can be described by a truth table as

$\overbrace{x_{11}\cdots x_{1n}}^{X_1}$	$\overbrace{x_{21}\cdots x_{2n}}^{X_2}$	...	$\overbrace{x_{m1}\cdots x_{mn}}^{X_m}$	$\overbrace{f_1\cdots f_n}^{F}$
0 ... 0	0 ... 0		0 ... 0	$f_{10}\cdots f_{n0}$
0 ... 0	0 ... 0		0 ... 0	$f_{11}\cdots f_{n1}$
⋮	⋮		⋮	⋮
1 ... 1	1 ... 1		1 ... 1	$f_{12^{mn}}\cdots f_{n2^{mn}}$

Table 1

By Theorem 2(a),

$$F(P_1,\ldots P_m)[X_1^{P_1}(X_1^{\widetilde{P}_1})\ldots(X_1^{\overbrace{\widetilde{P}_1}^{(n-1)}})]\ldots[X_m^{P_m}(X_m^{\widetilde{P}_m})\ldots X_m^{\overbrace{\widetilde{P}_m}^{(n-1)}}]$$

$$=\begin{cases}F(P_1,\ldots,P_m) & \text{if } X_1=P_1,\ \ldots,\ X_m=P_m\\ 0 & \text{otherwise}\end{cases}$$

Since the summation of Eq. (1) is over all possible values of $P_1,\ldots,P_m$, the above equation ensures that the function value of F of each row of the truth table is described by one and only one term of Eq. (1). Therefore, Eq. (1) describes not only precisely the truth table of F (thus the function F), but also in the compact form.

Theorem 6(b) may be proven by using Theorem 2(b) and a similar argument.

Example 1 Consider a simple vector Boolean function described in Table 2.

X_1	X_2	F
0	0	1
0	1	3
1	2	2
2	3	1
Otherwise		0

(a) A multivalued logic truth table

X_1	X_2	F
00	00	01
00	01	11
01	10	10
10	11	01
Otherwise		00

(b) The binary coded truth table of (a)

Table 2

The canonical sum-of-products form of this function is

$$F(X_1,X_2) = (01)X_1^{00}(\widetilde{X_1^{00}})(\widetilde{\widetilde{X_1^{00}}})\cdot X_2(\widetilde{X_2^{00}})(\widetilde{\widetilde{X_2^{00}}})$$
$$+ (11)X_1^{00}(\widetilde{X_1^{00}})(\widetilde{\widetilde{X_1^{00}}})\cdot X_2^{01}(\widetilde{X_2^{01}})(\widetilde{\widetilde{X_2^{01}}})$$
$$+ (10)X_1^{01}(\widetilde{X_1^{01}})(\widetilde{\widetilde{X_1^{01}}})\cdot X_2^{10}(\widetilde{X_2^{10}})(\widetilde{\widetilde{X_2^{10}}})$$
$$+ (01)X_1^{10}(\widetilde{X_1^{10}})(\widetilde{\widetilde{X_1^{10}}})\cdot X_2^{11}(\widetilde{X_2^{11}})(\widetilde{\widetilde{X_2^{11}}}) \qquad (3)$$

V. Application

Since the advent of integrated circuit digital components, many basic digital design philosophies have been changed. For example, the switching function minimization is now no longer to be important in the design procedure. On the other hand, digital design using ready-made components, SSI, MSI, and LSI has become today's standard procedure of designing digital systems.

Equations (1) and (2) suggest another possible method of designing digital systems using "component modules". The generalized components involved in these equations may be realized by using EXCLUSIVE-OR gates, since

$$X^P = X \oplus \overline{P} = X \oplus P \oplus I$$

where $X_1 \oplus X_2 = \overline{X}_1X_2+X_1\overline{X}_2$. For example, the first term of Eq. (3) expressed in terms of EXCLUSIVE-OR operation is

$$(01)(X_1 + \overline{00})(\widetilde{X_1 + \overline{00}})(\widetilde{\widetilde{X_1 + \overline{00}}})(X_2 + \overline{00})(\widetilde{X_2 + \overline{00}})(\widetilde{\widetilde{X_2 + \overline{00}}})$$

A general products term of Eq. (1),

$$F(P_1\ldots P_m)[(X_1 \oplus \overline{P}_1)\ldots\overbrace{(X_1 \oplus \overline{P}_1)}^{(n-1)}]\ldots[(X_m \oplus \overline{P}_m)\ldots\overbrace{(X_m \oplus \overline{P}_m)}^{(n-1)}]$$

may be realized by the gate circuit shown in Fig. 2(a) which may be symbolically represented by the block diagram shown in Fig. 2(b) in which the symbol ↷ denoted that the order of the ordered wires is rotated. The complete realization of Eq. (1) may then be presented by the diagram given in Fig. 3. For example, the realization of Eq. (3) is shown in Fig. 4. The realization of the product-of-sums canonical form of Eq. (2) may be derived in a similar way. Moreover, the method may be easily extended to sequential circuit design in which the next-state vector function and the output vector function are first represented in canonical form and then realized by ready-made circuit modules.

VI. Conclusion

A generalization of Boolean algebra into a more general algebraic structure that may be applied to the design of digital systems at a system level has been presented. Accordingly, DeMorgan's theorem, Shannon's theorem, and the expansion theorm have been generalized into more general forms. Two canonical forms, the sum-of-products and the product-of-sums, of any Boolean function in the generalized Boolean algebra have been found. These canonical forms suggested systematic realization schemes for logical design using some standard ready-made integrated-circuit digital component modules.

Hence the process of designing a digital system for a logical design may be automated by using these synthesis schemes. This method is applicable to both combinational and sequential logical designs.

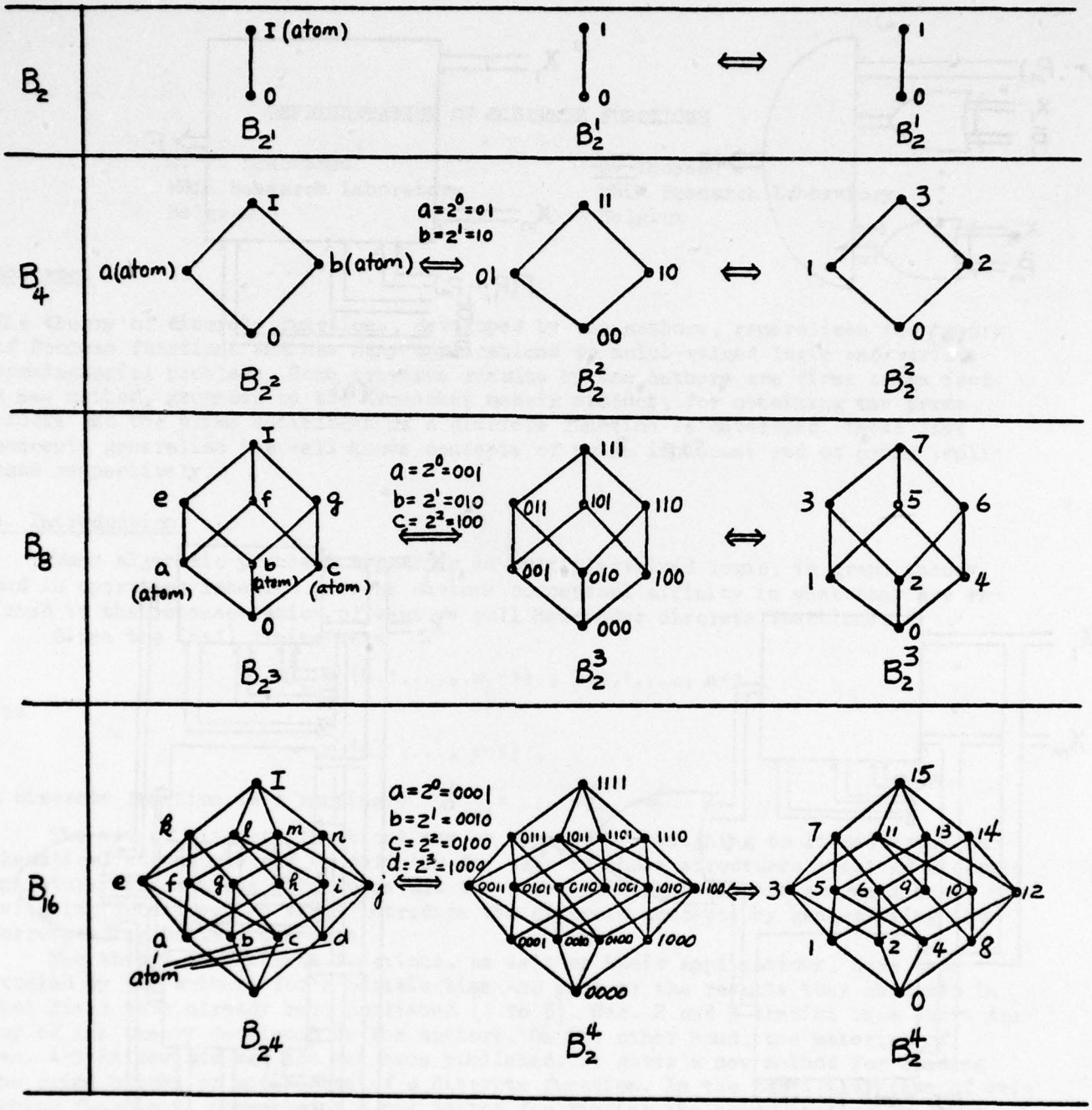
References

1. Computer, special issue on Multiple-Valued Logic, September 1974.

2. Abbott, J. C., Sets, Lattices, and Boolean Algebras, Allyn and Bacon, Inc., 1969.

3. Flegg, H. G., Boolean Algebra and Its Applications, John Wiley & Sons, Inc.,1964.

4. Braddock, R. C., Epstein, H. Yamanaka, "Multiple-Valued Logic Design and Applications in Binary Computers", 1971 Symposium on Multiple-Valued Logic Design, Buffalo, N. Y.


B2
I (atom)
0
B2¹
1
0
B2¹
1
0
B2¹
B4
I
a(atom)
b(atom)
0
B2²
a=2⁰=01
b=2¹=10
11
01
10
00
B2²
3
1
2
0
B2²
B8
I
e
f
g
a (atom)
b (atom)
c (atom)
0
B2³
a=2⁰=001
b=2¹=010
c=2²=100
111
011
101
110
001
010
100
000
B2³
7
3
5
6
1
2
4
0
B2³
B16
I
k
l
m
n
e
f
g
h
i
j
a
b
c
d
atom
B2⁴
a=2⁰=0001
b=2¹=0010
c=2²=0100
d=2³=1000
1111
0111
1011
1101
1110
0011
0101
0110
1001
1010
1100
0001
0010
0100
1000
0000
B2⁴
15
7
11
13
14
3
5
6
9
10
12
1
2
4
8
0
B2⁴

(a) (b) (c)

Fig. 1

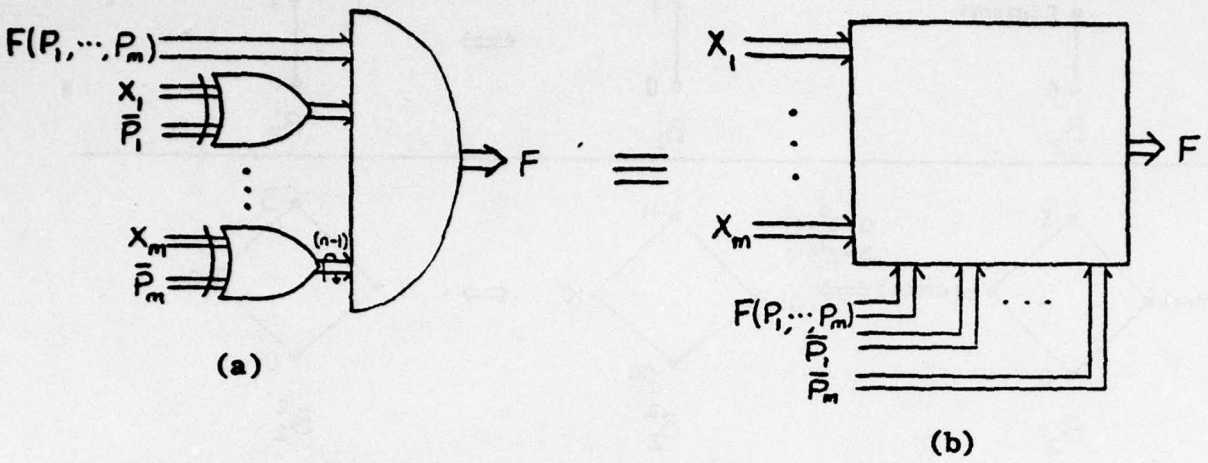


Fig. 2

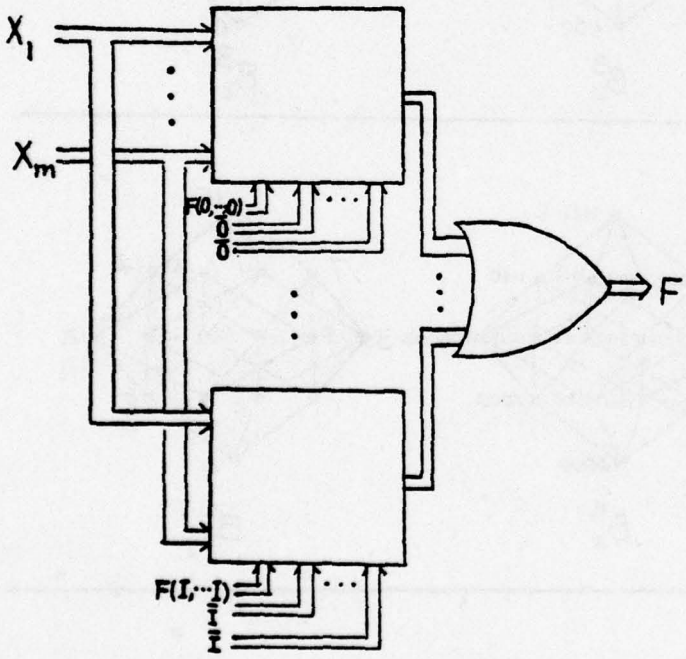


Fig. 3

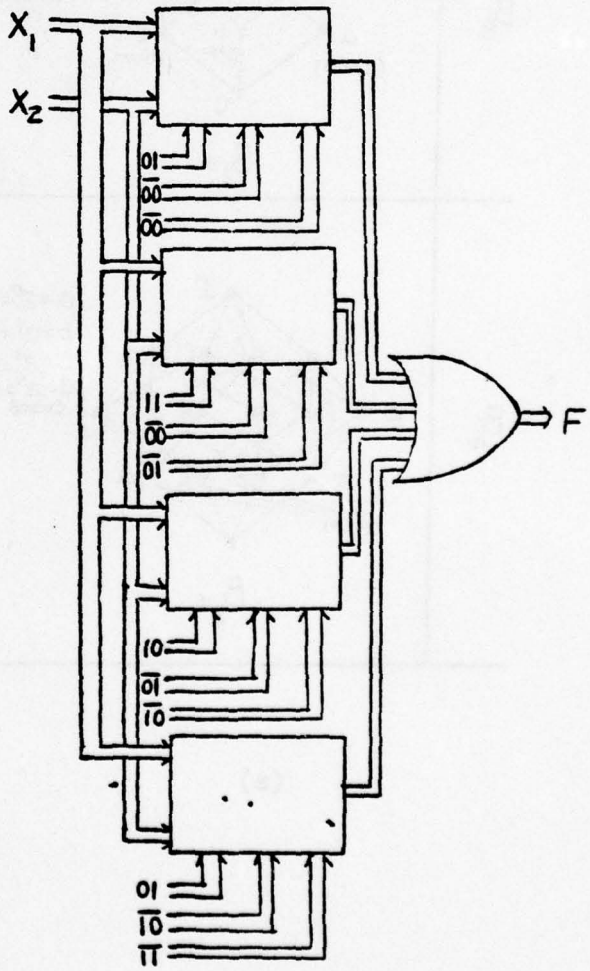


Fig. 4

REPRESENTATION OF DISCRETE FUNCTIONS

J. P. Deschamps[1]
MBLE Research Laboratory
Belgium

A. Thayse[1]
MBLE Research Laboratory
Belgium

Abstract

The theory of discrete functions, developed by the authors, generalizes the theory of Boolean functions and has many applications to multi-valued logic and various combinatorial problems. Some previous results by the authors are first summarized. A new method, grounded on the Kronecker matrix product, for obtaining the prime blocks and the prime antiblocks of a discrete function is developed. These last concepts generalize the well-known concepts of prime implicant and of prime implicate respectively.

1 roduction.

any algebraic problems appearing in multiple-valued logic, in graph theory and in operation research present obvious conceptual affinity in what they are related to the representation of what we call hereunder discrete functions.

Given the (n+1) finite sets

$$\mathcal{S}_i = \{0,1,\ldots, m_i-1\} \ , \ i=0,1,\ldots, n-1 \ ,$$

and

$$\mathcal{L} = \{0,1,\ldots, r-1\} \ ,$$

a discrete function is a mapping $F : \mathcal{S}_{n-1} \times \ldots \times \mathcal{S}_0 \rightarrow \mathcal{L}$.

The set of discrete functions may be studied by assigning to it different mathematical structures and by defining for each of these structures a set of discrete operators. Since these structures and the corresponding operators are well known for switching functions, we shall introduce the discrete concepts by generalizing the corresponding Boolean concepts.

The theory of discrete functions, as well as their applications, have been studied by the authors for a certain time and most of the results they obtained in that field have already been published [1 to 8]. Sec. 2 and 3 consist in a short survey of the theory developed by the authors. On the other hand, the material of sec. 4-5 is new and has not yet been published. It gives a new method for finding the prime blocks or antiblocks of a discrete function. In the particular case of switching functions, one obtains a new method for finding the prime implicants, that is competitive with other classical methods from the complexity point of view.

2. Algebraic structures for discrete functions.

The switching functions are classically studied and represented by using either the algebraic structure of Boolean lattice or the structure of Boolean ring. Similarly, there exists a lattice theory and a ring theory for discrete functions. These two theories have been extensively developed in [1] and [2], respectively. Hereunder follows a summary of the main obtained results.

---

[1] The authors are with the MBLE Research Laboratory, 2, avenue Van Becelaere, B-1170, Brussels, Belgium.

2.1. The lattice structure.

Any chain like $\mathcal{L}$ has the structure of distributive lattice if one defines the disjunction $\vee$ as the max. operation and the conjunction $\wedge$ as the min. operation. Furthermore, define the following exponentiation:

given a variable x_i and a subset C_i of $\mathcal{S}_i$, the discrete function $\tilde{x}_i^{(C_i)}$ is defined by

$$\tilde{x}_i^{(C_i)} = r-1 \text{ iff } x_i \in C_i \ ,$$

$$= 0 \text{ otherwise.}$$

Thanks to these notions, any discrete function admits the two following canonical expansions :

$$F(\underline{x}) = \bigvee_{\underline{e}} (F(\underline{e}) \wedge \tilde{x}_{n-1}^{(e_{n-1})} \wedge \ldots \wedge \tilde{x}_0^{(e_0)}) \ , \quad 0 \leq e_i \leq m_i - 1$$

and

$$F(\underline{x}) = \bigwedge_{\underline{e}} (F(\underline{e}) \vee \tilde{x}_{n-1}^{(\overline{e_{n-1}})} \vee \ldots \vee \tilde{x}_0^{(\overline{e_0})}) \ , \quad 0 \leq e_i \leq m_i - 1$$

where e_i holds for the singleton $\{e_i\}$, and $\overline{e}_i$ for $\mathcal{S}_i \setminus \{e_i\}$.
Moreover, $\underline{e}$ holds for the vector $(e_{n-1}, \ldots, e_1, e_0)$.
These two expansions give F as disjunction of minterms, or as conjunction of maxterms.

For switching functions exist more condensed but not canonical representations based upon the computation of the prime implicants or of the prime implicates of the given function. In order to define prime implicants and prime implicates for discrete functions we have first to generalize the notions of product term and sum term ; this can be done in three ways :

(1) a cube function [1,10] is a discrete function c of the type

$$c(x_{n-1}, \ldots, x_0) = \ell \wedge \tilde{x}_{n-1}^{(C_{n-1})} \wedge \ldots\ldots \wedge \tilde{x}_0^{(C_0)} \ ,$$

where $\ell \in \mathcal{L}$ and $C_i \subseteq \mathcal{S}_i$, i=0,...,n-1. Similarly, an anticube function is a discrete function of the type

$$a(x_{n-1}, \ldots, x_0) = \ell \vee \tilde{x}_{n-1}^{(C_{n-1})} \vee \ldots\ldots \vee \tilde{x}_0^{(C_0)} \ .$$

A cube (anticube) function smaller (greater) than F is a cube (anticube) of F ; a maximal cube (minimal anticube) of F is a prime cube (prime anticube) of F.

(2) A block function [1] is a cube function for which each set C_i is an interval i.e. $C_i = [a_i, a_i \oplus 1, \ldots, a_i \oplus h_i = b_i]$, where $\oplus$ is the sum mod. m_i and $0 \leq h_i \leq m_i - 1$; it is denoted under the form

$$\ell \wedge \tilde{x}_{n-1}^{[a_{n-1}, b_{n-1}]} \wedge \ldots\ldots \wedge \tilde{x}_0^{[a_0, b_0]} \ .$$

Similarly, an antiblock function is an anticube function for which each set C_i is an interval. The notions of (prime) block and (prime) antiblock of a discrete function

are then clearly defined.
(3) A convex block function [1,11] is a block function for which each interval $[a_i,b_i]$ is convex i.e. $a_i \leq b_i$. A convex antiblock function is an antiblock function such that the complement of each interval $[a_i,b_i]$ with respect to δ_i is a convex interval i.e. either $a_i > b_i$ or $a_i = 0$ or $b_i = m_i - 1$. Again, the notions of (prime) convex block and (prime) convex antiblock of a discrete function are clearly defined.

As an example, consider the function defined by table I ; it admits three prime cubes :

$$2 \wedge \tilde{x}^{(0,3,4)} \wedge \tilde{y}^{(0,1,3,4)} \quad , \quad 2 \wedge \tilde{x}^{(0,2,3,4)} \wedge \tilde{y}^{(1,3)} \quad , \quad 1 \wedge \tilde{x}^{(1)} \wedge \tilde{y}^{(2)} \quad ;$$

four prime blocks : $2 \wedge \tilde{x}^{[3,0]} \wedge \tilde{y}^{[3,1]}$, $2 \wedge \tilde{x}^{[2,0]} \wedge \tilde{y}^{[1,1]}$, $2 \wedge \tilde{x}^{[2,0]} \wedge \tilde{y}^{[3,3]}$, $1 \wedge \tilde{x}^{[1,1]} \wedge \tilde{y}^{[2,2]}$;

seven prime convex blocks : $2 \wedge \tilde{x}^{[3,4]} \wedge \tilde{y}^{[3,4]}$, $2 \wedge \tilde{x}^{[3,4]} \wedge \tilde{y}^{[0,1]}$, $2 \wedge \tilde{x}^{[0,0]} \wedge \tilde{y}^{[3,4]}$, $2 \wedge \tilde{x}^{[0,0]} \wedge \tilde{y}^{[0,1]}$, $2 \wedge \tilde{x}^{[2,4]} \wedge \tilde{y}^{[1,1]}$, $2 \wedge \tilde{x}^{[2,4]} \wedge \tilde{y}^{[3,3]}$, $1 \wedge \tilde{x}^{[1,1]} \wedge \tilde{y}^{[2,2]}$;

seven prime anticubes : $0 \vee \tilde{x}^{(0,3,4)} \vee \tilde{y}^{(1,2,3)}$, $1 \vee \tilde{x}^{(0,2,3,4)}$, $0 \vee \tilde{x}^{(0,2,3,4)} \vee \tilde{y}^{(2)}$, $0 \vee \tilde{x}^{(1)} \vee \tilde{y}^{(0,1,3,4)}$, $0 \vee \tilde{x}^{(0,1,3,4)} \vee \tilde{y}^{(1,3)}$, $1 \vee \tilde{y}^{(0,1,3,4)}$, $1 \vee \tilde{x}^{(0,3,4)} \vee \tilde{y}^{(1,3)}$;

five prime antiblocks : $0 \vee \tilde{x}^{[3,0]} \vee \tilde{y}^{[1,3]}$, $1 \vee \tilde{x}^{[2,0]}$, $0 \vee \tilde{x}^{[2,0]} \vee \tilde{y}^{[2,2]}$, $0 \vee \tilde{x}^{[1,1]} \vee \tilde{y}^{[3,1]}$, $1 \vee \tilde{y}^{[3,1]}$;

eight prime convex antiblocks : $0 \vee \tilde{x}^{[3,0]} \vee \tilde{y}^{[0,3]}$, $0 \vee \tilde{x}^{[3,0]} \vee \tilde{y}^{[1,4]}$, $1 \vee \tilde{x}^{[2,0]}$, $0 \vee \tilde{x}^{[2,0]} \vee \tilde{y}^{[0,2]}$, $0 \vee \tilde{x}^{[2,0]} \vee \tilde{y}^{[2,4]}$, $0 \vee \tilde{x}^{[0,1]} \vee \tilde{y}^{[3,1]}$, $0 \vee \tilde{x}^{[1,4]} \vee \tilde{y}^{[3,1]}$, $1 \vee \tilde{y}^{[3,1]}$.

Several methods have been described in [1] and [10] for finding the prime cubes, blocks, anticubes, antiblocks and convex antiblocks. A new method for finding the prime blocks and antiblocks will be given in sec.5.

Table I.

		x				
		0	1	2	3	4
y	0	0	2	2	0	0
	1	0	2	0	0	0
	2	2	1	2	2	2
	3	0	2	0	0	0
	4	0	2	2	0	0

Having found for instance the prime cubes of a given discrete function F, one can search for an irredundant cover of F by means of prime cubes and obtain a representation of F. In any way, all the representations (canonical or not) that can be obtained are of one of the two following types :

$$F = \bigvee_{k=1}^{m} (\ell_k \wedge \tilde{x}_{n-1}^{(C_{n-1,k})} \wedge \ldots \wedge \tilde{x}_0^{(C_{0,k})}) = \bigwedge_{j=1}^{q} (\ell'_j \vee \tilde{x}_{n-1}^{(C'_{n-1,j})} \vee \ldots \vee \tilde{x}_0^{(C'_{0,j})}).$$

They yield realizations of discrete functions by means of gates that perform the disjunction (max.), the conjunction (min.) and the choosen exponentiation.

We can define the negation of a discrete function G as follows :

$$\bar{G} = (r-1) - G \ .$$

With this definition of negation the De Morgan's laws hold, and F can also be written under the form

$$F = N_m(N_n(\ell_1, \tilde{x}_{n-1}^{(C_{n-1},1)}, \ldots, \tilde{x}_0^{(C_0,1)}), \ldots, N_n(\ell_m, \tilde{x}_{n-1}^{(C_{n-1},m)}, \ldots, \tilde{x}_0^{(C_0,m)})),$$

where the i-th Nand operator is defined as follows :

$$N_i(g_1, g_2, \ldots, g_i) = \overline{g_1 \wedge g_2 \wedge \ldots \wedge g_i} \ .$$

One obtains in this way a realization of F by means of gates that perform the Nand operations and the exponentiation.

An other interesting remark is that thanks to the canonical representations in minterms and maxterms, several classical theorems of complexity theory can be generalized [12].

We conclude this section by giving some other examples of applications of the theoretical material developed above. In some cases indeed, a discrete function can be used for describing in a condensed form the working of some digital system. If the fact for two arguments of the discrete function to be adjacent has no particular meaning for the described system, the concept of cube will be used. Suppose for instance that one wishes to realize an electrical dipole controlled by two switches, one with four positions and the other with five positions, according to the following rule : the dipole is a short circuit for the positions of the switches to which corresponds the value 1 in table II ; it is an open circuit for the positions to which corresponds the value 0. The problem is solved by finding the prime cubes of the binary discrete function of table II. The resulting dipole is given in figure 1.

Table II.

		switch 2				
		0	1	2	3	4
switch 1	0	0	0	0	0	0
	1	0	1	0	1	1
	2	0	0	1	1	0
	3	0	1	1	1	1

If the ring structure of each $\mathcal{S}_i$ has a particular meaning for the described system, the concept of block will be used : it is the case when analysing the transient behaviour of combinational and sequential networks [5,6].

Finally emphasis can be put on the chain structure of each $\mathcal{S}_i$ by using the concept of convex block ; it serves for the study and synthesis of partially symmetric switching functions [11].

2.2. The ring structure.

One can confer to the set of integers $=\{0,1,\ldots,r-1\}$ the structure of ring by defining the ring sum $\oplus$ as the sum mod.r and the ring product, denoted by a dot or by the absence of symbol, as the product mod.r. Define now the following exponentiation :

$$x_i^{(C_i)} = 1 \text{ iff } x_i \in C_i \ ,$$

$$= 0 \text{ otherwise}, \ \forall \ C_i \subseteq \mathcal{S}_i \ , \ \forall \ i=0,\ldots,n-1 \ .$$

In the case where $\mathcal{C}_i$ is the interval $[a_i, a_i \oplus 1, \ldots, a_i \oplus h_i = b_i]$, one simply write : $x_i^{[a_i,b_i]}$. Thanks to these notions, any discrete function admits the following canonical expansion ($\bigoplus$ represents the ring sum) :

$$F(\underline{x}) = \bigoplus_{\underline{e}} F(\underline{e})\, x_{n-1}^{(e_{n-1})} \ldots x_0^{(e_0)} ,$$

where $\underline{0} \leq \underline{e} = (e_{n-1}, \ldots, e_0) \leq \underline{m-1} = (m_{n-1}-1, \ldots, m_0-1)$.

In the next section, two other canonical expansions will be given, that generalize the classical Reed-Muller expansion.

We conclude this section by the following remark : in the case where $m_{n-1} = \ldots = m_0 = r = p$, with p a prime, any discrete function F is a mapping from $(GF(p))^n$ into $GF(p)$ and admits a representation as polynomial of degree at most (p-1) in each variable x_i i.e. under the form

$$F(\underline{x}) = \bigoplus_{\underline{e}} \ell(\underline{e})\, x_{n-1}^{e_{n-1}} \ldots x_0^{e_0} .$$

3. Differences of discrete functions.

Suppose that T represents one of the three following binary operations defined on $\mathcal{L}$: either the conjunction $\wedge$, or the disjunction $\vee$, or the mod.2 substraction $-$. The simple T-difference of F with respect to x_i, that we temporarily denote $\frac{TF}{Tx_i}$, is defined as follows :

$$\frac{TF}{Tx_i} = F(x_{n-1}, \ldots, x_i \oplus 1, \ldots, x_0) T F(x_{n-1}, \ldots, x_i, \ldots, x_0) ,$$

where $\oplus$ is the sum mod. m_i. The multiple T-differences are defined by induction :

$$\frac{T^{k_i} F}{Tx_i} = \frac{T}{Tx_i} \left(\frac{T^{k_i - 1} F}{Tx_i} \right) ,$$

and

$$\frac{T^{\underline{k}_0} F}{T\underline{x}_0} = \frac{T^{k_{p-1}}}{Tx_{p-1}} \left(\ldots \left(\frac{T^{k_1}}{Tx_1} \left(\frac{T^{k_0} F}{Tx_0} \right) \right) \right) ,$$

with $\underline{k}_0 = (k_{p-1}, \ldots, k_1, k_0)$ and $\underline{x}_0 = (x_{p-1}, \ldots, x_1, x_0)$.

In the case where T represents the conjunction, the T-differences are called meet diffences and denoted $\frac{p^{\underline{k}_0} F}{p\underline{x}_0}$. In the case where T represents the disjunction, the differences are called join differences and denoted $\frac{q^{\underline{k}_0} F}{q\underline{x}_0}$. Finally, when T represants the substraction mod.r, the T-differences are called ring differences and denoted $\frac{\Delta^{\underline{k}_0} F}{\Delta\underline{x}_0}$

3.1. Meet and join differences.

An important property of the meet and join differences is given by the

following theorem in which $\underline{m}_0-\underline{1}=(m_{p-1}-1,\ldots,m_1-1, m_0-1)$:

Theorem 3.1. [1]. (a) The meet difference $\frac{p^{\underline{m}_0-\underline{1}}F}{px_0}$ is the largest function smaller than F and independent of x_0. It is the disjunction of all the prime implicants (cubes, blocks or convex blocks) independent of x_0.

(b) Dual statement.

An algorithm for computing the prime blocks (antiblocks) of a discrete function and based upon the computation of the meet (join) differences has been described in [1]. Furthermore a more specific algorithm for switching functions has been developed [9]. In sec.5 of this paper a new algorithm will be presented.

3.2. Ring differences.

The properties of the ring differences are very similar to those of the classical finite differences in numerical analysis. The most important of these properties are gathered in theorems 3.2 and 3.3 below (Π represents the ring product)

Theorem 3.2. [2] (a) $$\frac{\Delta^{\underline{k}_0}F}{\Delta\underline{x}_0^{\underline{k}_0}} = \bigoplus_{\underline{e}_0} (-1)^{\sum_{j=0}^{p-1}(k_j+e_j)} \left[\prod_{j=0}^{p-1}\binom{k_j}{e_j}\right] F(\underline{x}_0 \oplus \underline{e}_0, \underline{x}_1) ,$$

with $$\underline{0}\leq\underline{e}_0=(e_{p-1},\ldots, e_0) \leq \underline{m}_0-\underline{1}=(m_{p-1}-1,\ldots,m_0-1),$$

where $\binom{k_j}{e_j}$ holds for the binomial coefficient evaluated mod.r.

(b) (Newton expansion)

$$F(\underline{x}_0,\underline{x}_1) = \bigoplus_{\underline{e}_0} \left[\prod_{i=0}^{p-1}\binom{x_i-h_i}{e_i}\right] \left(\frac{\Delta^{\underline{e}_0}F}{\Delta\underline{x}_0^{\underline{e}_0}}\right)_{\underline{x}_0=\underline{h}_0}, \quad \underline{0}\leq\underline{e}_0\leq\underline{m}_0-\underline{1} ,$$

with $\underline{h}_0=(h_{p-1},\ldots,h_0)$ and each (x_i-h_i) is computed mod.m_i.

In view of writing the Nyquist expansion introduce the following notation :

$$\underline{e}_0^{[\underline{1},\underline{m}_0-\underline{1}]} = (e_{p-1}^{[1,m_p-1]}, \ldots, e_0^{[1,m_0-1]}) .$$

Theorem 3.3. (Nyquist expansion) [2]

$$F(\underline{x}_0,\underline{x}_1) = \bigoplus_{\underline{e}_0} \left[\prod_{i=0}^{p-1} (x_i-h_i)^{[e_i,m_i-1]}\right] \left(\frac{\Delta^{\underline{e}_0^{[\underline{1},\underline{m}_0-\underline{1}]}}F}{\Delta\underline{x}_0^{[\underline{1},\underline{m}_0-\underline{1}]}}\right)_{\underline{x}_0=\underline{e}_0\oplus\underline{h}_0-\underline{e}_0}$$

Examples of Newton and Nyquist expansions can be found in [2].

It is clear that both Newton and Nyquist expansions could lead to the realization of multiple-valued networks provided that the appropriate building blocks were available ; from this point of view, the Nyquist expansion seems more appropriate since the exponentiation $x_i^{[e_i,m_i-1]}$ is easily performed by a sort of threshold gate [13].

Another generalization of the Reed-Muller expansion has been proposed by Kodandapani [14] ; the coefficients of that expansion are related to the sensitivities

of the function rather than to its ring differences [15].

Conclude this section by the case of functions from $(GF(p))^n$ into GF(p). With each such function F can be associated one and only one polynomial $\phi(F)$ the degree of which is at most (p-1) in each variable, and conversely each polynomial ϕ defines a function $f(\phi)$. Furthermore, for each variable x_i one can define a partial derivation operator $\frac{\partial}{\partial x_i}$ acting formally on each polynomial, and thus also a partial derivation operator acting on each function by the simple rule :

$$\frac{\partial F}{\partial x_i} = f\left(\frac{\partial \phi(F)}{\partial x_i}\right) .$$

The multiple derivatives are then defined by induction.

We can now state the following theorem [2].

Theorem 3.5. (a) $$\frac{\partial F}{\partial x_j} = (p-1) \bigoplus_{e=1}^{p-1} \frac{F(x_{n-1},\ldots,x_j \oplus e, \ldots, x_0) - F(x_{n-1},\ldots,x_j,\ldots,x_0)}{e} .$$

(b) (Taylor expansion) .

$$F(\underline{x} \oplus \underline{h}) = \bigoplus_{\underline{e}} \left(\frac{\partial^{\underline{e}} F}{\partial \underline{x}} \right)_{\underline{x}=\underline{h}} \frac{x_{n-1}^{e_{n-1}} \cdots x_0^{e_0}}{(i_{n-1}!)\ldots(i_0!)} ,$$

where

$$\underline{0} \leq \underline{e}=(e_{n-1},\ldots,e_0) \leq (p-1,\ldots,p-1) .$$

Let $S_i^{(k)}$ and $\mathcal{S}_i^{(k)}$ be the Stirling numbers [16] of the first and of the second type, respectively. The following theorem relates the ring differences and the derivatives.

Theorem 3.6. [8] (a) $$\frac{\Delta^k F}{\Delta x_j} = \bigoplus_{i=1}^{p-1} \frac{\mathcal{S}_i^{(k)}\, k!}{i!} \frac{\partial^i F}{\partial x_j^i} , \qquad k \leq p-1 ;$$

(b) $$\frac{\partial^k F}{\partial x_j^k} = \bigoplus_{i=1}^{p-1} \frac{S_i^{(k)}\, k!}{i!} \frac{\Delta^i F}{\Delta x_j^i} , \quad k \leq p-1.$$

The last theorem yields for functions on GF(p) the counterpart of classical relations in numerical analysis [17] , what confirms the relevance of the choosen definitions for differences and derivatives.

4. The extended vectors.

Let $[F(\underline{e})]$ be the truth vector of $F(\underline{x})$, with $\underline{e}=e_{n-1},\ldots e_1,e_0$; $0 \leq e_j \leq m_j-1$ $\forall j$. The partial truth vector of F with respect to $x_i \in \underline{x}$, that is : $(F(x_i=0) , F(x_i=1),\ldots, F(x_i=m_i-1))$ will be denoted $[F(e_i)]$ and the partial truth vector of F with respect to $\underline{x}_0 \subseteq \underline{x}$ will be denoted $[F(\underline{e}_0)]$. The T-partial extanded vector of discrete functions (F,...,G) with respect to a variable x_i will be denoted $\underline{\phi}_{x_i}^{(T)}(F,\ldots,G)$ and is defined as follows (T holds for the disjunction or the conjunction) :

$$\underline{\phi}_{\underline{x}_i}^{(T)}(F,\ldots,G)=\left([F(e_i)],\left[\frac{TF}{Tx_i}(e_i)\right],\ldots,\left[\frac{T^{m_i-2}F}{Tx_i}(e_i)\right],\frac{T^{m_i-1}F}{Tx_i},\ldots,\right.$$

$$\left.[G(e_i)],\left[\frac{TG}{Tx_i}(e_i)\right],\ldots,\left[\frac{T^{m_i-2}G}{Tx_i}(e_i)\right],\frac{T^{m_i-1}G}{Tx_i}\right)$$

It is recalled that $T^{m_i-1}F/Tx_i$ is independent of x_i. Clearly, in view of the above definition, $\underline{\phi}_{\underline{x}_i}^{(T)}(F)$ associates to $F : \mathcal{S} \to \mathcal{L}$, $m_i(m_i-1)+1$ applications :

$$\mathcal{S}_{n-1} \times \ldots \times \mathcal{S}_{i+1} \times \mathcal{S}_{i-1} \times \ldots \times \mathcal{S}_0 \to \mathcal{L}.$$

The T-partial extended vector of discrete functions $(F,\ldots,G)$ with respect to the variables in $\underline{x}_0 = x_{p-1}, x_{p-2}, \ldots, x_0 \subseteq \underline{x}$ will be denoted $\underline{\phi}_{\underline{x}_0}^{(T)}(F,\ldots,G)$ and is defined as follows :

$$\underline{\phi}_{\underline{x}_0}^{(T)}(F,\ldots,G) = \underline{\phi}_{\underline{x}_0}^{(T)}\left(\underline{\phi}_{\underline{x}_1}^{(T)}\left(\ldots\underline{\phi}_{\underline{x}_{p-1}}^{(T)}(F,\ldots,G)\right)\right)$$

The T-extended vector $\underline{\phi}^{(T)}(F,\ldots,G)$ derives finally from the above definitions when the complete set $\underline{x}$ of variables is considered, that is : $\underline{\phi}^{(T)}(F,\ldots,G) = \underline{\phi}_{\underline{x}}^{(T)}(F,\ldots,G)$.

The above definitions evidently hold whatever the representation of the discrete functions F,...,G may be. In particular it is important to note that they may be given as literal expressions or by means of their truth vector. Consider e.g. a function F given by means of its truth- or of its partial truth-vector. Let us define the following matrices :

I_i is the $m_i \times m_i$ unit matrix

P_i is the $m_i \times m_i$ primitive circulant, that is (only the 0's on the diagonal have been indicated) :

$$P_i = \begin{bmatrix} 0 & & & & 1 \\ 1 & 0 & & & \\ & 1 & & & \\ & & & & \\ & & & 0 & \\ & & & 1 & 0 \end{bmatrix}$$

E_i is the $m_i \times 1$ matrix of 1's, that is (t indicates the transpose operation) :

$$E_i = [1,1,\ldots,1]^t .$$

$M_i^{(v)}$ is the $m_i \times (m_i(m_i-1)+1)$ matrix of 0's and of $(r-1)$'s defined as follows :

$$M_i^{(v)} = (r-1) \times \left[I_i, (I_i+P_i), (I_i+P_i+P_i^2), \ldots, \left(I_i + \sum_{j=1}^{m_i-2} P_i^j\right), E_i\right]$$

($\times$ denotes here the real product).

The dual matrix, that is $M_i^{(\wedge)}$ is obtained from $M_i^{(v)}$ by interchanging the 0's with the

$(r-1)$'s. Both types of matrices will be denoted by the single symbol $M_i^{(T)}$. Let $(T,\perp)$ represent a matrix product operator with the additive law T and the multiplicative law $\perp$. The Kronecker matrix product [18] with the multiplicative law T will be denoted $\overset{T}{\otimes}$; in view of the extended vector definition, one has for T the disjunction or the conjunction and with $\perp$ the correspondent dual operation :

$$\underset{\sim}{\phi}_{x_i}^{(T)}(F) = [F(e_i)]\ (T,\perp)\ [M_i^{(T)}]$$

$$\underset{\sim}{\phi}^{(T)}(F) = [F(\underline{e})]\ (T,\perp)\ \Big[\overset{\perp}{\underset{i=n-1,0}{\otimes}} M_i^{(T)}\Big]$$

Let us give some comments about the above formulas ; let $(r-1)\times X_i^{(v)}$ be a submatrix of $M_i^{(v)}$ and $\underline{y}_i^{(v)}$ be a subvector of $\underset{\sim}{\phi}_{x_i}^{(v)}$; one has : $[F(e_i)]\ (T,\perp)\ [(r-1)\times X_i^{(v)}] = \underline{y}_i^{(v)}$ for the following values of $X_i^{(v)}$ and $\underline{y}_i^{(v)}$:

$$X_i^{(v)} \longrightarrow y_i^{(v)}$$

$$I_i = \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix} \longrightarrow [F(e_i)]$$

$$I_i + P_i = \begin{bmatrix} 1 & & & & 1 \\ 1 & 1 & & & \\ & 1 & 1 & & \\ & & 1 & \ddots & \\ & & & & 1 \end{bmatrix} \longrightarrow \left[\frac{qF}{qx_i}(e_i)\right]$$

$$I_i + \sum_{j=1}^{p-2} P_i^j = \begin{bmatrix} 1 & 0 & 1 & & 1 \\ 1 & 1 & 0 & & 1 \\ 1 & 1 & 1 & & 1 \\ \cdot & 1 & 1 & \cdot & 1 \\ \cdot & \cdot & \cdot & \cdot & \\ 1 & \cdot & \cdot & & 0 \\ 0 & 1 & 1 & & 1 \end{bmatrix} \longrightarrow \left[\frac{q^{m_i-2}F}{qx_i}(e_i)\right]$$

$$E_i = [1\ 1\ 1\ \ldots\ 1]^t \longrightarrow \frac{q^{m_i-1}F}{qx_i}$$

$\underset{\sim}{\phi}_{x_i}^{(T)}(F)$ for T the disjunction, derives immediately from the above matricial relations ; a dual relation holds for T the conjunction. $\underset{\sim}{\phi}^{(T)}(F)$ immediately derives from the Kronecker matrix product with multiplicative law T whose definition is briefly recalled below. Consider e.g. two matrices M and R of order m and n respectively ; moreover let m_{ij} be the element of M located at the intersection of the ith row and of the jth column ; one has then by definition :

$$\overset{T}{M} \otimes R = \begin{bmatrix} m_{oo}\ ^T R & m_{o1}\ ^T R & \cdots & m_{o(r-1)}\ ^T R \\ m_{10}\ ^T R & m_{11}\ ^T R & \cdots & m_{1(r-1)}\ ^T R \\ \cdots & \cdots & \cdots & \cdots \\ m_{(r-1)o}\ ^T R & m_{(r-1)1}\ ^T R & \cdots & m_{(r-1)(r-1)}\ ^T R \end{bmatrix}$$

where $m_{ij}\ ^T R$ is the matrix R "multiplied" by the scalar m_{ij}. The iteration scheme proposed by the Kronecker matrix product is then exactly the same as the iteration which holds for the definition of the extended vector.

The Boolean case

From the above definitions one deduces that the T-partial extended vector of binary functions F,...,G with respect to a variable x_i is :

$\underset{\sim}{\phi}^{(T)}_{x_i}(F,\ldots,G)=(F(x_i=0),\ F(x_i=1),\ \frac{TF}{Tx_i},\ \ldots,\ G(x_i=0),\ G(x_i=1),\ \frac{TG}{Tx_i})$

$\underset{\sim}{\phi}^{(T)}_{x_i}(F)$ associates thus to F : $B_2^n \to B_2$ three applications : $B_2^{n-1} \to B_2$.

The matrices $M_i^{(T)}$ reduce to the following ones :

$$M_i^{(\vee)} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \ , \ M_i^{(\wedge)} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

The following relation holds between $\vee$- and $\wedge$-extended vectors :

$$\underset{\sim}{\phi}^{(F)}\ F(\underset{\sim}{e}) = \overline{\underset{\sim}{\phi}^{(\perp)}\ [\overline{F}(\underset{\sim}{e})]}$$

where $\overline{\ }$ holds for the Boolean complementation.

The extended vector has 3^n components and, from its definition, it appears that it may be partitioned into three subvectors of equal length that are the extended vectors of the subfunctions $F(x_{n-1}=0)$, $F(x_{n-1}=1)$ and of the T-difference TF/Tx_{n-1}.

5. Prime blocks and prime antiblocks of a discrete function.

Let us denote by $(\tilde{x}_i^{[j,j \oplus k]})$ the vector

$(\tilde{x}_i^{[0,k]}\ ,\ \tilde{x}_i^{[1,k \oplus 1]},\ \tilde{x}_i^{[2,k \oplus 2]},\ \ldots,\ \tilde{x}_i^{[m_i-1,\ k \oplus m_i-1]})$

The vectors $\underset{\sim}{K}_i^{(T)}$ are then defined as follows :

$\underset{\sim}{K}_i^{(\wedge)}=((\tilde{x}_i^{[j,j \oplus 0]}),\ (\tilde{x}_i^{[j,j \oplus 1]}),\ \ldots,\ (\tilde{x}_i^{[j,j \oplus (m_i-2)]})\ ,\ \tilde{x}_i^{[j,j \oplus (m_i-1)]}\ _{=r-1})^t$

$\underset{\sim}{K}_i^{(\vee)} = (r-1)-\underset{\sim}{K}_i^{(\wedge)}$

$\underset{\sim}{K}^{(T)}$ is the Kronecker product : $\underset{\sim}{K}^{(T)}= \overset{T}{\underset{i=n-1,0}{\otimes}}\ \underset{\sim}{K}_i^{(T)}$.

Theorem 5.1.

The blocks (antiblocks) of the function $F(\underset{\sim}{x})$ are the terms of :

$$[\underset{\sim}{\phi}^T(F)]\ (\perp,T)\ \underset{\sim}{K}^{(T)}$$

with T the conjunction (disjunction) and $\perp$ the dual operation.

Proof For n=1, m=3, a routine verification shows that :

(a) $[F(0),F(1),F(2),F(0)F(1),F(1)F(2),F(2)F(0),F(0)F(1)F(2)]\ (\vee,\wedge)$

$[\tilde{x}^{[0,0]}, \tilde{x}^{[1,1]}, \tilde{x}^{[2,2]}, \tilde{x}^{[0,1]}, \tilde{x}^{[1,2]}, \tilde{x}^{[2,0]}, \tilde{x}^{[0,2]}, r-1]^t =$

$[F(0)\tilde{x}^{[0,0]} \vee F(1)\tilde{x}^{[1,1]} \vee F(2)\tilde{x}^{[2,2]} \vee F(0)F(1)\tilde{x}^{[0,1]} \vee F(1)F(2)\tilde{x}^{[1,2]} \vee$

$F(2)F(0)\tilde{x}^{[2,0]} \vee F(0)F(1)F(2)$

(b) $[F(0)F(1),F(2),F(0) \vee F(1),F(1) \vee F(2),F(2) \vee F(0),F(0) \vee F(1) \vee F(2)]\ (\wedge,\vee)$

$[\tilde{x}^{[1,2]}, \tilde{x}^{[2,0]}, \tilde{x}^{[0,1]}, \tilde{x}^{[2,2]}, \tilde{x}^{[0,0]}, \tilde{x}^{[1,1]}, 0]^t =$

$(F(0) \vee \tilde{x}^{[1,2]})(F(1) \vee \tilde{x}^{[2,0]})(F(2) \vee \tilde{x}^{[0,1]})(F(0) \vee F(1) \vee \tilde{x}^{[2,2]})$

$(F(1) \vee F(2) \vee \tilde{x}^{[0,0]})(F(2) \vee F(0) \vee \tilde{x}^{[1,1]})(F(0) \vee F(1) \vee F(2))$

The proof is easily achieved by perfect induction on m and on n.

The prime blocks and the prime antiblocks of F are then obtained from the above theorem by deleting in the expressions the blocks smaller and the antiblocks greater than other ones respectively. It is obvious that theorem 5.1 immediately leads to an easily programmable algorithm for obtaining the list of the prime blocks and of the prime antiblocks of a discrete function. The complexity of this algorithm is fixed and is of $\prod_i [m_i(m_i-1)+1]$.

An elementary illustrative example is treated here below ; the function considered is a ternary function of a binary variable x_0 and of a ternary variable x_1, that is :

$$F = \tilde{x}_0^{(1)}\tilde{x}_1^{(0)} \vee \tilde{x}_0^{(0)}\tilde{x}_1^{(0)} \vee \tilde{x}_0^{(0)}\tilde{x}_1^{(1)} \vee 1\tilde{x}_0^{(0)}\tilde{x}_1^{(2)} \vee 1\tilde{x}_0^{(1)}\tilde{x}_1^{(2)}$$

The truth vector of that function is

$$[F(e_1,e_0)] = (2, 2, 2, 0, 1, 1)$$

The matrix $M^{(\wedge)}$ is :

$$M^{(\wedge)} = M_1^{(\wedge)} \overset{\vee}{\otimes} M_0^{(\wedge)}$$

$$= \begin{bmatrix} 0 & 2 & 2 & & 0 & 2 & 0 & & 0 \\ 2 & 0 & 2 & & 0 & 0 & 2 & & 0 \\ 2 & 2 & 0 & & 2 & 0 & 0 & & 0 \end{bmatrix} \overset{\vee}{\otimes} \begin{bmatrix} 0 & 2 & 0 \\ 2 & 0 & 0 \end{bmatrix} =$$

$$\begin{bmatrix} 0 & 2 & 0 & 2 & 2 & 2 & 2 & 2 & 0 & 2 & 0 & 2 & 2 & 2 & 0 & 2 & 0 & 0 & 2 & 0 \\ 2 & 0 & 0 & 2 & 2 & 2 & 2 & 2 & 2 & 0 & 0 & 2 & 2 & 2 & 2 & 0 & 0 & 2 & 0 & 0 \\ 2 & 2 & 2 & 0 & 2 & 0 & 2 & 2 & 2 & 0 & 2 & 0 & 0 & 2 & 0 & 2 & 2 & 2 & 0 & 2 & 0 \\ 2 & 2 & 2 & 2 & 0 & 0 & 2 & 2 & 2 & 2 & 0 & 0 & 2 & 0 & 0 & 2 & 2 & 2 & 2 & 0 & 0 \\ 2 & 2 & 2 & 2 & 2 & 2 & 0 & 2 & 0 & 2 & 2 & 2 & 0 & 2 & 0 & 0 & 2 & 0 & 0 & 2 & 0 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 0 & 0 & 2 & 2 & 2 & 2 & 0 & 0 & 2 & 0 & 0 & 2 & 0 & 0 \end{bmatrix}$$

The extended vector $\underline{\phi}^{(\wedge)}(F)$ is :

$$\underline{\phi}^{(\wedge)}(F) = [F(e_1,e_0)]\ (\wedge,\vee)\ M^{(\wedge)}$$

$$= (2\ 2\ 2\ 2\ 0\ 0\ 1\ 1\ 1\ 2\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 0)$$

The vector $\underline{K}^{(\wedge)}$ is :

$$\underline{K}^{(\wedge)} = \underline{K}_1^{(\wedge)} \hat{\otimes} \underline{K}_o^{(\wedge)}$$

$$= [(\tilde{x}_1^{(o)}, \tilde{x}_1^{(1)}, \tilde{x}_1^{(2)}, \tilde{x}_1^{[0,1]}, \tilde{x}_1^{[1,2]}, \tilde{x}_1^{[2,0]}, 2) \hat{\otimes}$$
$$(\tilde{x}_o^{(o)}, \tilde{x}_o^{(1)}, 2)]^t$$

$$= (\tilde{x}_1^{(o)}\tilde{x}_o^{(o)}, \tilde{x}_1^{(o)}\tilde{x}_o^{(1)}, \tilde{x}_1^{(o)}, \tilde{x}_1^{(1)}\tilde{x}_o^{(o)}, \tilde{x}_1^{(1)}\tilde{x}_o^{(1)}, \tilde{x}_1^{(1)},$$
$$\tilde{x}_1^{(2)}\tilde{x}_o^{(o)}, \tilde{x}_1^{(2)}\tilde{x}_o^{(1)}, \tilde{x}_1^{(2)}, \tilde{x}_1^{[0,1]}\tilde{x}_o^{(o)}, \tilde{x}_1^{[0,1]}\tilde{x}_o^{(1)}, \tilde{x}_1^{[0,1]},$$
$$\tilde{x}_1^{[1,2]}\tilde{x}_o^{(o)}, \tilde{x}_1^{[1,2]}\tilde{x}_o^{(1)}, \tilde{x}_1^{[1,2]}, \tilde{x}_1^{[2,0]}\tilde{x}_o^{(o)}, \tilde{x}_1^{[2,0]}\tilde{x}_o^{(1)},$$
$$\tilde{x}_1^{[2,0]}, \tilde{x}_o^{(o)}, \tilde{x}_o^{(1)}, 2]^t$$

The expression of F as a disjunction of blocks is thus :

$$F = \tilde{x}_1^{(o)}\tilde{x}_o^{(o)} \vee \tilde{x}_1^{(o)}\tilde{x}_o^{(1)} \vee \underline{\tilde{x}_1^{(o)}} \vee \tilde{x}_1^{(1)}\tilde{x}_o^{(o)} \vee 1\tilde{x}_1^{(2)}\tilde{x}_o^{(o)} \vee 1\tilde{x}_1^{(2)}\tilde{x}_o^{(1)} \vee$$
$$\underline{1\tilde{x}_1^{(2)}} \vee \underline{\tilde{x}_1^{[0,1]}\tilde{x}_o^{(o)}} \vee 1\tilde{x}_1^{[1,2]}\tilde{x}_o^{(o)} \vee 1\tilde{x}_1^{[2,0]}\tilde{x}_o^{(o)} \vee 1\tilde{x}_1^{[2,0]}\tilde{x}_o^{(1)} \vee$$
$$\underline{1\tilde{x}_1^{[2,0]}} \vee \underline{1\tilde{x}_o^{(o)}}$$

The prime blocks in the above expression have been underlined.
The prime antiblocks are obtained by making use of dual types of computations.
The above computations, using the vectors $\underline{\phi}^T$ and $\underline{K}^T$ and the matrix M are easily programmable ; for hand computations the above algorithm becomes quickly cumbersome when the number and the size of the variables increase. Several improvements (which are specially interessant for Boolean functions) of this algorithm will be proposed in a further paper by the authors [19] which will appear in "Discrete Mathematics". The computation of prime convex blocks and of prime convex antiblocks are also obtained in this text by using Kronecker matrix products.

References

[1] J. P. Deschamps and A. Thayse, On a theory of discrete functions, Part I, The lattice structure of discrete functions, Phil. Res. Repts., 28, pp. 397-423, 1973.

[2] A. Thayse and J. P. Deschamps, On a theory of discrete functions, Part II, The ring and field structures of discrete functions, Phil. Res. Repts., 28, pp. 424-465, 1973.

[3] J. P. Deschamps, On a theory of discrete functions, Part III, Decomposition of discrete functions, Phil. Res. Repts., 29, pp. 193-213, 1974.

[4] A. Thayse, On a theory of discrete functions, Part IV, Discrete functions of functions, Phil. Res. Repts., 29, pp. 305-329, 1974.

[5] J. P. Deschamps and A. Thayse, Applications of discrete functions, Part I, Transient analysis of combinational networks, Phil. Res. Repts., 28, pp. 497-529, 1973.

[6] A. Thayse, Applications of discrete functions, Part II, Transient analysis of asynchronous switching networks, Phil. Res. Repts., 29, pp. 155-192, 1974.

[7] A. Thayse, Applications of discrete functions, Part III, The use of functions of functions in switching circuits, Phil. Res. Repts., 29, pp. 429-452, 1974.

[8] A. Thayse, Differential calculus for functions from $(GF(p))^n$ into GF(p), Phil. Res. Repts., 29, pp. 560-586, 1974.

[9] A. Thayse, Disjunctive and conjunctive operators for Boolean functions, Phil. Res. Repts., 28, pp. 1-6, 1973.

[10] M. Davio and G. Bioul, Representation of lattice functions, Phil. Res. Repts., 25, pp. 370-388, 1970.

[11] J. P. Deschamps, Partially symmetric switching functions, Phil. Res. Repts., 28, pp. 245-264; 1973.

[12] M. Davio, Unpublished notes on complexity.

[13] Z. Vranesic and K. Smith, Engineering aspects of multi-valued logic systems, Computer, 9, pp. 34-41, 1974.

[14] K. L. Kodandapani, Logic realizations based on Reed-Muller canonical forms, Thesis, Indian Institute of Science, Bangalore, 1974.

[15] J. P. Deschamps, Unpublished notes.

[16] M. Abramowitz and I. Stegun, Handbook of mathematical function, US National Bureau of Standards, 1964.

[17] Ch. J. de la Vallée Poussin, Cours d'analyse infinitésimale , tome I, Gauthier-Villars, Paris, 1959.

[18] R. Bellman, Introduction to matrix analysis, Mc Graw-Hill, 1960

[19] A. Thayse, Difference operators and extended truth vectors for discrete functions, Discrete mathematics, to appear.

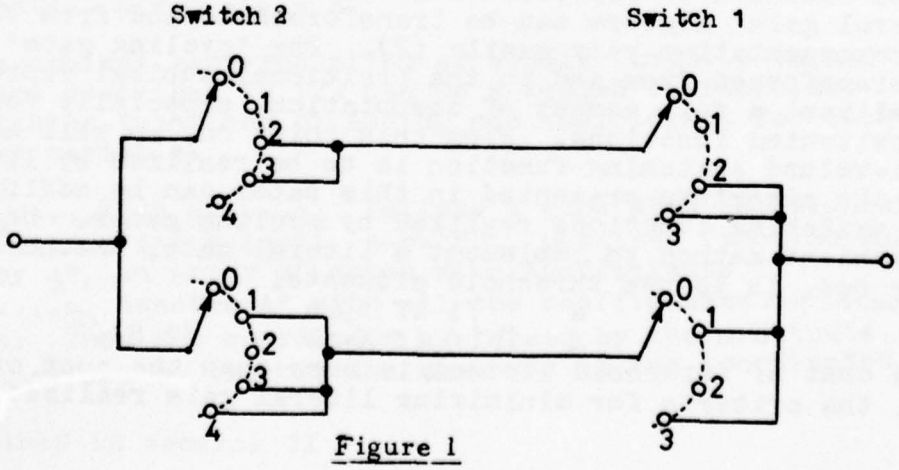


Figure 1

A COMPUTER-ORIENTED HEURISTIC MINIMIZATION ALGORITHM FOR MULTIPLE-OUTPUT MULTI-VALUED SWITCHING FUNCTIONS

Peter T. Cheung
Packard Instrument Inc.
Downers Grove
Ill. U. S. A.

Dave M. Purvis
Packard Instrument Inc.
Downers Grove
Ill. U. S. A.

ABSTRACT

A computer-oriented heuristic algorithm for the minimization of multiple-output, multi-valued switching functions is described. The positional cubical representation for multi-valued switching functions is extended to represent multiple-output functions. The algorithm generates a near minimum implicant solution without generating all possible prime implicants. This algorithm is especially well suited for minimizing the literal gates (generated by threshold elements) realization of multi-valued switching functions with many input and output variables. Some strategies used are also applicable for minimization of Boolean functions.

I. INTRODUCTION

In this paper we attempt to solve two main problems in the minimization of multiple-output, multi-valued switching functions. The first problem is to determine the criteria for minimization of multiple-output, multi-valued switching functions. The second problem is how to minimize functions with many input and output variables.

There are basically two main families of algebra for multi-valued switching functions that can be implemented easily. One family of algebra uses the literal gates (2,6)

$$\overset{a\ b}{X} = \begin{cases} p-1 & \text{if } a \leqslant X \leqslant b \\ 0 & \text{otherwise} \end{cases}$$

Another family of algebra uses the cycling gates (7) $X^{\underline{Y}} = (X+Y) \bmod p$. The positional cubical representation for multi-valued switching functions is easily stored and processed by a digital computer (1,2) and can be extended to represent multiple-output switching functions. The 'literal gate' algebra can be transformed to and from the positional cubical representation very easily (2). The 'cycling gate' algebra can also be transformed from and to the positional cubical representation but not without a fair amount of computation, especially for minimization of switching functions. From this point on, we will assume that the multi-valued switching function is to be realized by literal gates, although the algorithm presented in this paper can be modified to minimize switching functions realized by cycling gates. Note that the most convenient method to implement a literal gate, whether complemented or not, is to use threshold elements,

$$\overset{a}{X} = \begin{cases} 1 & \text{if } X \geqslant a \\ 0 & \text{if } X < a \end{cases} \quad (2,8).$$

Since the cost of threshold elements is more than the cost of AND and OR gates, the criteria for minimizing literal gate realized multi-

valued switching functions will be the number of threshold elements required. If two solutions require the same number of threshold elements to implement, then the one that requires the least number of AND and OR gates should be chosen.

The computer storage requirement and computing time, in general, increase exponentially with the number of input and output variables, and also with the number of logical levels (1-5). Hong, Cain and Ostapko (1) used a special case of the positional cubical representation for multi-valued switching functions (2) and designed a heuristic approach for minimizing Boolean switching functions. In this paper, a new strategy is introduced which decomposes the switching function into smaller disconnected arrays of cubes[1]. Instead of minimizing a large array of cubes, the minimization is done on many smaller disconnected subarrays of cubes. This preprocessing of the switching function can save a significant amount of storage requirement and computing time as the number of input and output variables increases. Almost all present algorithm experience difficulties in minimizing functions which have many disconnected subarrays of cubes (1). For example, Hong, Cain and Ostapko have successfully minimized several 30 input, 40 output Boolean functions with millions of mir-terms, but have failed to handle the 16-variable EXCLUSIVE OR function. The new strategy introduced decomposes the EXCLUSIVE OR function into 2^{15} subarrays, each having only one cube. This implies the function cannot be minimized further and the result is obtained almost immediately.

II. DEFINITION

The detailed definition of the algebraic representation and the positional cubical representation of multi-valued switching functions is described fully in previous papers (2). The following is a summary of the algebra:-

1. Let $L = \{0,1,.....,p-1\}$ be the set of logic values.
2. Let . (AND or PRODUCT) and + (OR or SUM) be binary functions on the set L such that for all a and b in L, $a \cdot b = \min(a, b)$ and $a + b = \max(a, b)$.
3. Literal gates:- $$a\,b_X = \begin{cases} p-1 & \text{if } a \leqslant X \leqslant b \\ 0 & \text{otherwise} \end{cases}$$ where X,a,b are in L and $a \leqslant b$.
4. Complement:- $\bar{X} = p - 1 - X$.

The cubical representation of single output p-valued switching functions described in (2) can be extended to represent multiple-output switching functions by adding one coefficient for each additional output variable.

$A^* = (K_a^1, K_a^2, \ldots, K_a^m)\underline{a}_1\underline{a}_2\ldots\underline{a}_n$ defines a cube with coefficients $K_a^i \in L$ for i=1,2,...,m. Any n-input, m-output, p-valued function can be expressed by $\underline{A}^*$, an array of cubes with coefficients. $A = \underline{a}_0\underline{a}_1\ldots\ldots\underline{a}_n$ denotes A^* with all the coefficients replaced by 1's and 0's. Thus the coordinate $\underline{a}_0$ formed by the m-outputs (coefficients) can be treated the same way as the coordinates for

1 to be defined in section II

the input variables. A* can be split up into a p-1 ON array and a don't care array as follows:-
$\underline{ON}_i$ = A* with all coefficients having value i replaced by 1, and all other coefficients replaced by 0. Delete cubes which have all their coefficients equal to 0.
DC = A* with all don't care coefficients replaced by 1 and all other coefficients replaced by 0. Delete cubes which have all their coefficients equal to 0.

Exanple 12: Given the function F shown in figure 1.

X_2 / X_1	0	1	2	3
0	2	2	0	0
1	3	3	0	0
2	0	0	1	1
3	0	0	d	d

Output Y_1

X_2 / X_1	0	1	2	3
0	d	d	0	0
1	3	3	0	0
2	0	0	d	d
3	0	0	3	3

Output Y_2

Figure 1. Function F - an example of multiple output, multi-valued switching function.

The function F is represented by the following array of cubes:-

$$\underline{F}^* = \begin{bmatrix} (2,d)(1000)(1100) \\ (3\ 3)(0100)(1100) \\ (1,d)(0010)(0011) \\ (d,3)(0001)(0011) \end{bmatrix}$$ where d is don't care

F* can be split up into p-1 ON arrays and a don't care array:-
The ON array for logical 3 :-

$$\underline{ON}_3 = \begin{bmatrix} (11)(0100)(1100) \\ (01)(0001)(0011) \end{bmatrix}$$

The ON array for logical 2 :-

$$\underline{ON}_2 = \begin{bmatrix} (10)(1000)(1100) \end{bmatrix}$$

The ON array for logical 1 :-

$$\underline{ON}_1 = \begin{bmatrix} (10)(0010)(0011) \end{bmatrix}$$

The DC array :-

$$\underline{DC} = \begin{bmatrix} (01)(1000)(1100) \\ (01)(0010)(0011) \\ (10)(0001)(0011) \end{bmatrix}$$

Definition 1 Two cubes $C=\underline{c}_0\underline{c}_1\cdots\underline{c}_n$ and $D=\underline{d}_0\underline{d}_1\cdots\underline{d}_n$ are disconnected if and only if there exists more than one coordinate k such that $\underline{c}_k \bigcap \underline{d}_k = \emptyset$. If two cubes are disconnected, then there is no one implicant that can cover part of both cubes and nothing else. In other words, if C and D are disconnected, then the set $\{X \mid X \text{ is a cube and } X \bigcap C \neq \emptyset,\ X \bigcap D \neq \emptyset, \text{ and } X = (C \bigcup D) \bigcap X\}$ is null.

Definition 2 Two arrays of cubes $\underline{A}$ and $\underline{B}$ are disconnected if and only if for every cube A in $\underline{A}$ and for every cube B in $\underline{B}$, cube A and cube B are disconnected. If two arrays of cubes are disconnected, there is no one implicant that can cover part of both arrays and nothing else. In other words, if $\underline{A}$ and $\underline{B}$ are disconnected arrays of cubes, then the set $\{X \mid X \text{ is a cube and } X \bigcap \underline{A} \neq \emptyset,\ X \bigcap \underline{B} \neq \emptyset, \text{ and } X = (\underline{A} \bigcup \underline{B}) \bigcap X\}$ is null.

III STRATEGIES

Strategy 1:-

From past experience, we found that the number of threshold elements needed, and the number of AND and OR gates needed, to realize a switching function usually decreases as the number of cubes in the cover for the switching function decreases. Thus the main strategy used in this algorithm is to reduce the number of cubes in the cover for the switching function, which will hopefully produce a cover that will reduce the number of threshold elements and AND and OR gates. Iterations are made based on a heuristic algorithm that will reduce the number of cubes in the cover; the best cover, however, is determined by the number of threshold elements and AND and OR gates used in the implementation and not the number of cubes. This method will produce a near minimum implicant solution.

Strategy 2:-

For a multiple-output, multi-valued switching function with many input and output variables, the function can usually be decomposed into many disconnected subarrays of cubes. Strategy 2 is based on the fact that if two arrays of cubes are disconnected, there is no one implicant that can cover part of both arrays and nothing else. This implies one can minimize the disconnected arrays of cubes separately without affecting the overall solution. Since the storage requirement and computing time necessary to minimize any array of cubes is in general exponentially proportional to the number of cubes in the array, the decomposition of a large array of cubes into smaller disconnected arrays of cubes and then minimizing them individually, will reduce the storage requirement and computing time significantly. The actual reduction of storage requirement and computing time depends only on how many disconnected arrays of cubes the function can be decomposed into and how many cubes each disconnected array has. It was found that for almost all other minimization algorithms, the more disconnected arrays of cubes the function can be decomposed to, the less efficient the algorithm is. By using the proposed strategy, the efficiency of most minimization algorithms is greatly increased for the worst case (which the algorithm may fail to minimize), although no improvement is made for the best case.

Other less important strategies used are described in section IV algorithm description.

IV ALGORITHM DESCRIPTION

Algorithm 1

Main procedure of a heuristic minimization algorithm for multiple output, multi-valued switching functions.

Step 1) Accept the definition of the function in cubical notation as an array of cubes $\underline{F}^*$.

Step 2) Calculate $\underline{ON}_i$ and $\underline{DC}_i$ for i = 1,2,...,p-1 :-
$\underline{ON}_i$ = $\underline{F}^*$ with all coefficients having value i replaced by 1 and all other coefficients replaced by 0. Delete cubes which have all their coefficients equal to 0.

$\underline{DC}_i$ = $\underline{F}^*$ with all coefficients having values of don't care, p-1,p-2,...,i+1 replaced by 1 and all other coefficients replaced by 0. Delete cubes which have all their coefficients equal to 0.

Step 3) Decompose $\underline{ON}_i$ into disconnected subarrays $\underline{ON}_i^1, \underline{ON}_i^2, \ldots, \underline{ON}_i^m$ for i = 1,2,...,p-1 (refer to algorithm 2 and 3).
Another possibility is to decompose $\underline{ON}_i \cup \underline{DC}_i$ into disconnected subarrays $\underline{ONDC}_i^1, \underline{ONDC}_i^2, \ldots, \underline{ONDC}_i^x$. Then $\underline{ON}_i^j$ = $\underline{ONDC}_i^j$ # $\underline{DC}_i$ and the corresponding don't care array $\underline{DC}_i^j$ = $\underline{ONDC}_i^j$ # $\underline{ON}_i$. This would produce a more optimum solution but requires more computing time. This decomposition can be used as a preprocessing procedure for any minimization algorithm.

Step 4) For each disconnected subarray $\underline{ON}_i^x$ with its associated don't care array $\underline{DC}_i$, use heuristic algorithm 4 to reduce the number of cubes in the subarray.

Step 5) The last step is to use algorithm 5 to further reduce the number of threshold elements and the number of AND and OR gates required by the solution. (Step 4 and Step 5 use basically the same heuristic strategies. The only difference is the criteria used for the iteration loop. Step 4 iterates on the reduction of the number of cubes. Step 5 iterates on the reduction of the actual number of threshold elements and the number of AND and OR gates. Step 4 produces an approximate solution in a reasonable amount of computing time and step 5 refines the approximate solution to a more optimum solution.).

Algorithm 2

Decompose $\underline{F}_i$, a general array of cubes, into disconnected subarrays:-
Given any array $\underline{C}$ with m cubes.

Step 1 Initialize i=1, j=1.

Step 2 Create a disconnected subarray S_j as follows :-

a. Initialize k=1, h=1, and $S_j=\emptyset$.

b. Remove C^1 from C and put it in S_j. Relabel the remaining cubes in C consecutively.

c. If the kth cube in S_j does not exist, go to step 3. (disconnected subarray S_j has been calculated).

d. If the hth cube in C does not exist, set k = k + 1, h = 1, and go to step 2c.

e. Otherwise compare the kth cube in S_j with the hth cube in C. If they are connected, remove the hth cube in C, add it to the end of S_j and relabel the remaining cubes in C consecutively and set h = h+1.

f. Go to step 2d.

Step 3 If C has no more cubes, go to step 5.

Step 4 Otherwise set j = j+ 1 , and go to step 2 to obtain the next disconnected subarray.

Step 5 The array C has been decomposed into disconnected subarray S_j's.

Note:- 1. The computing time of algorithm 2 is dominated by the testing of connected cubes. It can be shown that the maximum number of tests for connected cubes is (m-1)m/2.

2. In implementing this algorithm, no storage space for the subarrays are needed. All that is required is a flag for each cube to indicate which subarray it belongs to. Thus the net increase in core requirements is insignificant.

3. This algorithm is very general. The only requirement is that the switching function be represented by the positional cubical notation. The switching function can be binary or multi-valued, with single or multiple-output.

Algorithm 3

Decompose F_i, an array of minterm cubes, into disconnected subarrays.

Given any array C with m nim-term cubes.

Step 1 Initialize j = 1.

Step 2 Sort the array in ascending (or descending) order (treat each cube as a binary number).

Step 3 Compare all pairs of consecutive cubes in the array. If they are connected, label them to the same subarray.

Step 4 Set j=j+1. If j=n+1, go to step 7.

Step 5 Rotate the coordinates of each cube to the right (or to the left). That is rotate each cube $C = c_1, c_2, \cdots, c_{n+1}$ to $c_{(n+1)}, c_1, \cdots, c_n$.

Step 6 Go to step 2.

Step 7 The array C has been decomposed into disconnected subarrays.

Note :- The computing time for this algorithm is as follows :-

a. Sort the array n+1 times which requires

$$(n+1)\sum_{i=2}^{m}\log_2 i = (n+1)(m-1)(\log_2 m+1)/2$$

comparisons between cubes.

b. Rotate each one of m cubes in the array n times.

c. (m-1)(n+1) tests for connected cubes.

It can be shown that the computing time for algorithm 3 is proportional to m*n and not m^2 as in algorithm 2 (refer to Appendix A for an analysis of the computing time needed for the two algorithms).

Algorithm 4

Heuristic algorithm for reducing the number of cubes in any ON array with any DC array. This algorithm is essentially the same as steps M8-M16 of the heuristic algorithm described in [1]. Refer to [1] for detail description and explaination of the algorithm.

Step 1. Let F be any disconnected subarray ON_i^x and DC be DC_i.

Step 2 Let F be the distant one merging of $F \bigcup DC$

Step 3 Let $\overline{F} = U\,\textcircled{\#}\,F$. The disjoint sharp $\textcircled{\#}$ is the same as the regular # except that the resulting cubes are disjoint. Two cubes C and D are disjoint if and only if $C \bigcap D$ is null.

Step 4 Let $\overline{F}$ be $\overline{F}$ expanded against F.

Step 5 Generate disjoint F by $U\,\textcircled{\#}\,(\overline{F} \bigcup DC)$.

Step 6 Let F be F expanded against $\overline{F}$ and compute the number of cubes in F. This is the first solution for F. Steps 7 to 10 are the main loop which will produce a decreasing size solution for F.

Step 7 Reduce each cube in F against the other cubes in $F \bigcup DC$.

Step 8 Reshape F_i.

Step 9 Let F be F expanded against $\overline{F}$ and compute the solution size.

Step 10 If the size of the new solution is smaller than the solution immediately before the last execution of step 7, go to step 7. Otherwise F is the solution.

Algorithm 5

Heuristic algorithm for reducing the number of threshold elements and the number of AND and OR gates needed to implement the funcion.

Step 1 Initialize T, the set of threshold elements to $\emptyset$, and N, the number of AND and OR gates needed to 0.

Step 2 For logical level i = p-1,p-2,···,1 ;
a. Set DC = DC_i

b. Go down the list of cubes in $\underline{ON}_i$ and for each cube C,

1. Remove any interval that is not needed [2,8] (perform a getmin between C and DC).
2. Combine any remaining intervals if possible. Example: Try to combine the two intervals in (01010) into one interval as (01110)
3. Extend any interval to the end if possible. Example: Try to extend (01010) to either (11011) or (11010) or (01011).
4. Modify (either extend or reduce) any remaining intervals, if possible, to use threshold elements that are required by previous cubes. Example: If X^3 is used by previous cubes, then try to reduce (01100) to (00100) and extend (00011) to (00111).
5. Add any threshold elements needed to implement C to the set T and add the number of AND and OR gates needed to implement C to N.
6. Set DC to DC$\cup$C.

Step 3 If the newly obtained solution is better than the previous one, store the solution, the set of threshold elements required (T), and the number of AND and OR gates needed (N) for the solution. Otherwise, the previous stored solution is the solution, and the end of the algorithm.

Step 4 Compute another solution for the function by executing steps 7 to 9 of algorithm 4, and go to step 1.

APPENDIX A

The following is a discussion on the computing time needed to decompose an array into disconnected subarrays:-

Let a be the computing time needed to test for connection between two cubes, b be the computing time needed to compare the magnitude of two cubes, and c be the computing time needed to rotate a cube. The actual absolute value of a, b, and c depends on how the positional cubical notation is implemented in the computer and which computer is used. A rough estimate of their values, however, can be done as follows (assuming the number of logic values p is less than or equal to the number of bits in each computer word) :-

$$a = \begin{cases} (n+1) \text{ mask instructions plus} \\ (n+1) \text{ AND instructions plus} \\ (n+1) \text{ tests for 0} \end{cases}$$

b = x compare instructions

c = x rotate instructions

$$\text{where } x = \frac{p * \text{number of inputs} + \text{number of outputs}}{\text{number of bits in each word}}$$

From the definition of the two algorithms, it would be more efficient to expand all the cubes into min-term cubes and use algorithm 3 instead of algorithm 2 to decompose the array if

$$am_1(m_1-1)/2 > a(m-1)(n+1) + b(n+1)(m-1)(\log_2 m+1)/2 + cmn.$$

where m_1 is the number of cubes in the array,
m is the number of min-term cubes in the array if all the cubes are expanded.
n is the number of input variables.

REFERENCES

1. S,J.Hong,R.G.Cain,D.L.Ostapko,"Mini : A Heuristic Approach for Logic Minimization",*IBMJ.Res.Develop.* Sept. 1974.

2. Y.H.Su and P.T.Cheung,"Computer Minimization of Multi-valued Switching Functions", IEEE Trans. Computer, Sept. 1972.

3. R.G.Miller, SWITCHING THEORY , New York, Wiley, 1965.

4. M.A. Harrison, INTRODUCTION TO SWITCHING AND AUTOMATA THEORY, Macraw-Hill, 1965.

5. D.L.Dietmeyer, LOGIC DESIGN OF DIGITAL SYSTEM, Boston, Allyn and Bacon 1971.

6. C.M.Allen and D.D.Givone "Design of multi-valued logic system 1",Dept. EE,State Univ. New York, Buffalo. N.Y. Rep.dsl-66-1.

7. Z.G.Vranesic,E.S.Lee,and K.C.Smith, "A many-valued algebra for switching functions", IEEE Trans.Computer, Oct. 1970.

8. J.E.Birk and D.E.Farmer,"An Algebraic Method For Designing Multi-valued Logic Circuit Using Pfincipally Binary Components", submitted to IEEE Trans.Computer.

SYNTHESIS OF MULTIPLE-VALUED LOGIC NETWORKS BASED ON TREE-TYPE UNIVERSAL LOGIC MODULES

Tatsuo HIGUCHI and Michitaka KAMEYAMA
Department of Electronic Engineering, Faculty of Engineering,
Tohoku University, Aoba, Aramaki, Sendai, Japan

ABSTRACT

This paper presents a theoretical study on the synthesis of multiple-valued logic networks including binary logic ones based on tree-type universal logic modules(T-ULM's). The mathematical notation of T-ULM is introduced. The definition of some basic concepts of T-ULM, the mathematical properties of T-ULM's in regard to a canonical expansion, and various manipulations of a functional expression are discussed which have not been so far satisfactorily studied. On the basis of the mathematical properties, an algorithm for synthesizing any multiple-valued function of n variables with a smaller number of modules is presented. In this algorithm only true and constant inputs are allowed at input terminals. The algorithm is consisted of two parts: one is a functional decomposition. The other is the proper order of the expansion which is the problem of finding the most imcomplete tree structure. Thus it is established that the systematic and nonexhaustive procedure of the algorithm gives a suboptimal solution for the reduction of the number of T-ULM's.

I. INTRODUCTION

In the binary logic, the recent development of integrated circuit technology has made possible using the complex modules as building blocks in place of the NAND or NOR gates. These modules can realize any logic function of up to a fixed number of variables. Such modules are called universal logic modules(ULM's) [1-2].

In the multiple-valued logic, there are various type of operators such as in POST algebra which constructs a complete system [3]. However, the logic design is not so easy as in the binary logic. In addition to it, the number of the operators necessary to realize any logic function of n variables increases, particularly in the operators which construct a complete system only with a single gate such as Sheffer stroke functions [4]. Under this criteria, the use of ULM's in multiple-valued logic seems to be attractive in practical applications. The ternary T-gate which is called 'conditioned disjunction' by logicians is this kind of modules [5-8]. The modules such as T-gates are so called tree-type universal logic modules(T-ULM's) [2].

However, neither the mathematical properties of T-ULM's nor the minimamization procedure of the network of T-ULM's have not been studied satisfactorily so far. It is well known that the network of T-ULM's is realized in a tree structure. Yau and Tang [2] present the three possible methods for the reduction of the number of T-ULM's in binary logic, but its systematic and nonexhaustive procedure has not been obtained.

In this paper, we shall first introduce the mathematical notation of T-ULM which is the extention of the one of T-gate. Based on this notation, the mathematical properties of T-ULM's in regard to a canonical expansion, various manipulations of a functional expression, etc. are discussed. Then, an algorithm for synthesizing a logic network of any large number of variables is presented. This algorithm gives the suboptimal solution for the synthesis of a network of T-ULM's. In the algorithm, only true and constant inputs are allowed, because in the multiple -valued logic the number of unary functions is much larger than binary one. For an example in the ternary logic there exist $3^3 = 27$ unary functions. However, this algorithm will be extended also in the case where unary functions are allowed.

II. MATHEMATICAL PROPERTIES OF T-ULM'S

In the r-valued logic system, let the truth value be the element of a set $A = \{0, 1, \ldots, r-1\}$. If the T-ULM has k control variables, then the number of input/output terminals is $r^k + k + 1$. So we define T-ULM of k control variables as follows:

DEFINITION 1. Let $C = (c_1, c_2, \ldots, c_i, \ldots, c_k)$ be the control vector of T-ULM of which the elements are k control variables. The number of the distinct states of the vector C is r^k, and let the scalar S be the one to one mapping of the vector C, that is, $C \leftrightarrow S$ $(0 \le S \le r^k - 1)$, where $S = \sum_{i=1}^{k} c_i r^{i-1}$. Let the vector P be $(p_0, p_1, \ldots, p_j \ldots, p_{r^k-1})$. P is called the vector of residue functions hereafter. The output of T-ULM U_{out} is defined as

$U_{out} = U(p_0, p_1, \ldots, p_j, \ldots, p_{r^k-1} ; c_1, c_2, \ldots, c_i, \ldots, c_k)$
$= U(P ; C)$.
If $C \leftrightarrow S = j$, $U_{out} = p_j$.

The symbol of T-ULM is shown in Fig.1.

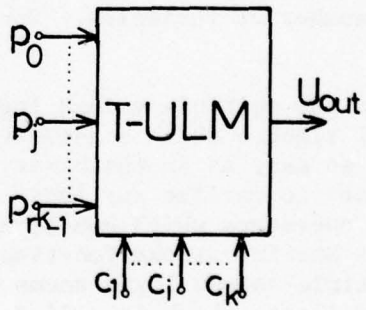


Fig.1 The symbol of T-ULM

EXAMPLE 1. $r = 2$, $k = 2$.
$U_{out} = U(p_0, p_1, p_2, p_3 ; c_1, c_2)$.
Then $U(P ; 0, 0) = p_0$, $U(P ; 1, 0) = p_1$,
$U(P ; 0, 1) = p_2$ and $U(P ; 1, 1) = p_3$.

$r = 3$, $k = 1$ (the case of ternary T-gate)
$U_{out} = U(p_0, p_1, p_2 ; c_1)$.
Then $U(P ; 0) = p_0$, $U(P ; 1) = p_1$
and $U(P ; 2) = p_2$.

THEOREM 1. Let a be the element of the set A. Given ℓ such that $\ell \in (1, \ldots, k)$. If $c_\ell = a$, then m_a is defined such that $m_a = \sum'_{i=1}^{k} c_i r^{i-1} + a r^{\ell-1}$ (the number of distinct values of m_a is r^{k-1}), where Σ' denotes the summation except $i = \ell$. Let $g(x)$ be a unary function in the r-valued logic. If $P_{m_a} = g(a)$ is hold for all the values of a, then $U(P ; C) = g(c_\ell)$.

PROOF. If $S = m_a = \sum_{i=1}^{k} c_i r^{i-1} + a r^{l-1}$, then $U(P ; C) = p_{m_a} = g(a)$. Since for all the value of a, the relation $U(P ; C) = g(a)$ is hold, $U(P ; C) = g(c_l)$.

EXAMPLE 2. $r = 3$, $k = 2$.
Let $c_1 = a$ and $m_a = c_2 3 + a$. If $a = 0$, $m_0 = 0, 3, 6$ according to $c_2 = 0, 1, 2$ respectively. If $a = 1$, $m_1 = 1, 4, 7$ according to $c_2 = 0, 1, 2$ respectively. If $a = 2$, $m_2 = 2, 5, 8$ according to $c_2 = 0, 1, 2$ respectively.
Therefore $U(0, 1, 2, 0, 1, 2, 0, 1, 2 ; c_1, c_2) = c_1$.

THEOREM 2. If the element of the vector P are all the same, i.e. $p_0 = p_1 = \ldots = p_j = \ldots = p_{r^k-1}$, then $U(P ; C) = p_0$ for all the value of c_i.
PROOF. By DEFINITION 1, the theorem is obvious.

THEOREM 3. Let the vector X be the set of n variables $(x_1, x_2, \ldots, x_n)$, and X_c be the vector of k variables in X. Then $U(p_0(X), \ldots, p_j(X), \ldots, p_{r^k-1}(X) ; X_c) = U(p_0(X', X_c \leftrightarrow 0), \ldots, p_j(X', X_c \leftrightarrow j), \ldots, p_{r^k-1}(X', X_c \leftrightarrow r^k-1) ; X_c)$, where $X' \cap X_c = \phi$(empty set), $X' \cup X_c = X$.
PROOF. If $X_c \leftrightarrow j$, then $U_{out} = p_j(X) = p_j(X', X_c \leftrightarrow j)$. This relation is hold for $j \in (0, 1, \ldots, r^k-1)$.

THEOREM 4. If constant and true inputs are allowed, an arbitrary function of n variables can completely be expressed with a maximum of $(r^{bk}-1)/(r^k-1)$ T-ULM's, where $b = \lfloor n/k \rfloor$ ($\lfloor x \rfloor$ denotes the smallest integer such that $\lfloor x \rfloor \geq x$).
PROOF. First consider a function of $n = n'k$ (n' is integer) variables. It is clear that if $n' = 1$, then the function can be expressed with one T-ULM. Assume that a function of $n'k$ variables can completely be expressed with $(r^{n'k}-1)/(r^k-1)$ T-ULM's. The function of $(n' + 1)k$ variables is generally expressed by THEOREM's 2, 3 as follows:

$$f(X) = U(f(X), f(X), \ldots, f(X), \ldots, f(X) ; X_c)$$
$$= U(p_0(X'), \ldots, p_j(X'), \ldots, p_{r^k-1}(X') ; X_c).$$

Each p_j is a function of $n'k$ variables, so $f(X)$ can be expressed with $(r^{n'k}-1) \cdot r^k/(r^k-1) + 1 = (r^{(n'+1)k} - 1)/(r^k-1)$ T-ULM's. Therefore by mathematical induction, for all the values of n' the theorem is proved. Since an arbitrary function of n variables is contained by the function of $\lfloor n/k \rfloor k = bk$ variables, it is completely expressed with $(r^{bk}-1)/(r^k-1)$ T-ULM's.

THEOREM 5. Canonical expansion theorem. An arbitrary function of n variables can always be expressed in the following form
$f(X) = U(p_0(X'), p_1(X'), \ldots, p_j(X'), \ldots, p_{r^k-1}(X') ; X_c)$
where $p_0(X') = f(X', X_c \leftrightarrow 0), \ldots, p_j(X') = f(X', X_c \leftrightarrow j), \ldots, p_{r^k-1}(X') = f(X', X_c \leftrightarrow r^k-1)$.
PROOF. It is obvious according to THEOREM's 2 and 3.

THEOREM 6. A function of n variables is expressed in a complete tree structure. Let U_1 be the T-ULM of the first level which is the nearest to the output terminal. Similarly let U_l be the T-ULM of the l-th level. Then the number of U_l's is $r^{k(l-1)}$, where $1 \leq l \leq \lfloor n/k \rfloor$.

PROOF. By THEOREM 4, the number of T-ULM's of up to the l-th level is $(r^{lk} - 1)/(r^k-1)$. The number of T-ULM's of up to the $l-1$ -th level is $(r^{(l-1)k} - 1)/(r^k-1)$. Then the number of the l-th level is $(r^{lk}-1)/(r^k-1)$ $-(r^{(l-1)k}-1)/(r^k-1) = r^{k(l-1)}$.

A function of n variables is expressed in a canonical tree structure as shown in Fig.2.

DEFINITION 2. In the r-valued logic, the complement of x is defined as

$x + \bar{x} = r$ (+ arithmetic sum).

THEOREM 7. $\overline{U(P\ ;\ C)} = U(\bar{P}\ ;\ C)$.

PROOF. If $C \leftrightarrow S = j$, then $U(P\ ;\ C) = p_j$. Therefore $\overline{U(P\ ;\ C)} = \bar{p}_j$. This implies that $\overline{U(P\ ;\ C)} = U(\bar{P}\ ;\ C)$.

THEOREM 8. Let g(x) be an arbitrary unary function. Then $g(U(P\ ;\ C)) = U(g(P)\ ;\ C)$.

PROOF. If $C \leftrightarrow S = j$, then $U(P\ ;\ C) = p_j$. Therefore $g(U(P\ ;\ C)) = g(p_j)$. This implies that $g(U(P\ ;\ C)) = U(g(P)\ ;\ C)$.

THEOREM 9. For each control variable c_i $(i = 1, \ldots, k)$ an unary function $g_i(x)$ is given. Then

$U(P\ ;\ g_1(c_1), g_2(c_2), \ldots, g_k(c_k))$

$= U(p_{1'}, p_{2'}, \ldots, p_{j'}, \ldots, p_{(r^k-1)'}\ ;C)$

where $j' = \sum_{i=1}^{k} (g_i(c_i))r^{i-1}$.

PROOF. If $S = j = \sum_{i=1}^{k} c_i r^{i-1}$, then

$g(C) \leftrightarrow S' = j' = \sum_{i=1}^{k} (g_i(c_i))r^{i-1}$. Since $U(P\ ;\ C) = p_j$, $U(P\ ;\ g(C)) = p_{j'}$.

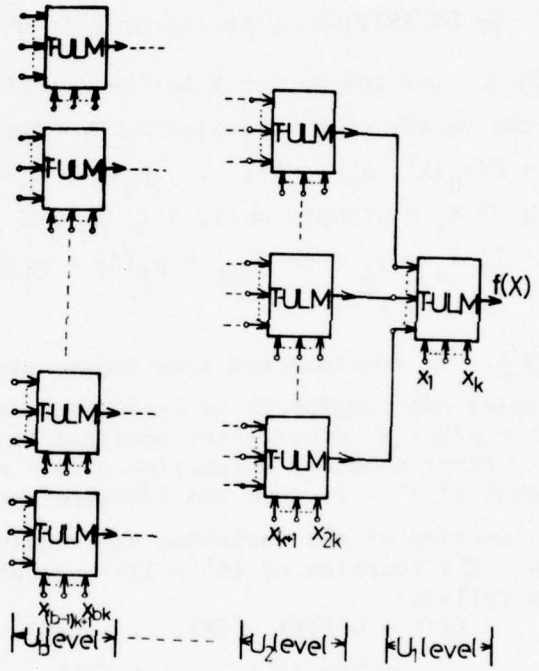


Fig. 2 Canonical tree structure

III. AN ALGORITHM FOR SYNTHESIZING A NETWORK OF T-ULM's

In this section, the attempt is made to reduce the number of T-ULM's in logic networks. Since THEOREM 4 assures functional completeness of the network of T-ULM's, any logic function of several variables is realized with the T-ULM's of few control variables in a tree structure. However, as the number of variables of a function becomes large, the number of T-ULM's necessary to realize the function exponentially increases in a canonical expression. We would like to present the

technique for the reduction of the number of T-ULM's.

1. Functional decomposition

DEFINITION 3. Let f(x) be a function of n variables, and let a set of n variables be divided into two disjunctive subsets Y and Z, where Y is composed of p variables ($1 \le p \le n-1$) and where $Y \cap Z = \phi$(empty set), $Y \cup Z = X$. We construct a decomposition table with the variables in Y defining the columns and the variables in Z defining the rows. Then the number of different vectors of the columns is defined as 'column multiplicity' i.e. K(Y ; f). The number of different vectors of the rows is defined as 'row multiplicity' i.e. K(Z ; f).

EXAMPLE 3. r = 3, k = 1
Consider the function $f(x_1, x_2, x_3)$ as shown in Table 1. If $Y = (x_1, x_2)$ and $Z = (x_3)$, then the different vectors of the columns are C_0, C_1 and C_2. Therefore, K(Y ; f) = 3. Similarly K(Z ; f) = 3.

	x_1, x_2 \ x_3	0	1	2
C_0	00	0	1	2
C_1	01	1	2	0
C_2	02	2	0	1
C_3	10	1	2	0
C_4	11	2	0	1
C_5	12	0	1	2
C_6	20	2	0	1
C_7	21	0	1	2
C_8	22	1	2	0

Table 1 The truth table of EXAMPLE 3

THEOREM 10. Let f(x) be decomposed as the following form
$f(X) = g(\alpha_1(Y), \alpha_2(Y), \ldots, \alpha_u(Y), Z)$. ·····(1)
The minimum value of integer u is given as $u \approx \lfloor \log_r K(Y ; f) \rfloor$.

If u is unity, Eq. (1) is called simple disjunctive decomposition. Eq.(1) corresponds to the block diagram of Fig. 3.

Now we consider how to select optimal Y, Z and whether the decomposition is allowed in respect to the cost of the given function. Let M_n be the upper bound of the number of T-ULM's necessary to realize the function of n variables. By Theorem 4, $M_n = (r^{\lceil n/k \rceil k} - 1)/(r^k - 1)$. In Eq.(1), g is a function of u + n - p variables. Each α_i (i = 1,...,u) is a function of p variables. Therefore the total upper bound on the cost of the function in Eq.(1) can be expressed as

$$F = u \cdot M_p + M_{u+n-p}$$
$$= u \cdot (r^{\lceil p/k \rceil k} - 1)/(r^k - 1) + (r^{\lceil (u+n-p)/k \rceil k} - 1)/(r^k - 1). \quad (2)$$

It is reasonable that Y and Z which minimize F in Eq.(2) are selected. However, if the minimum value of F is grester than M_n, then the decomposition is not effective in the reduction of the number of T-ULM's. From this point of view, we carry out the decomposition if $F \le M_n$ for the minimum F. For an allowed decomposition, we must consider how to decide the functions $\alpha_1, \alpha_2, \ldots, \alpha_u$.

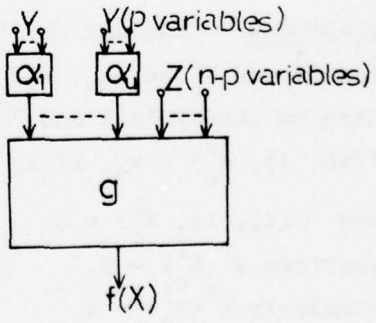


Fig. 3 The block diagram of the decomposition

If the vectors of the columns are the same, then in the corresponding columns the same code of length u is assigned, that is, the compatible sets are assigned the same code of $\alpha_1, \ldots, \alpha_u$. However it is difficult to estimate the number of

T-ULM's for the assignments (α_1, ..., α_u). Then we shall take one of the possible assignments. Once such a decomposifion has been allowed, the functions α_1, ..., α_u and g are each tested to determine if the decomposition is allowed. In this way, we carry out the decomposition iteratively until the decomposition is not allowed. This procedure is summarized in Subalgorithm 1.

Subalgorithm 1.
(1) Divide the given n variables into two disjunctive subsets Y and Z.
(2) Calculate the column multiplicity K(Y ; f) and u = $\lfloor \log_r K(Y ; f) \rfloor$.
(3) Calculate F of Eq.2.
(4) After all the possible pairs of Y and Z are checked, minimum F is selected.
(5) If $F \leq M_n$, go to (6). If $F > M_n$, the decomposition is not allowed, then go to Subalgorithm 2.
(6) Assign the same code to the compatible sets respectively. The functions α_1, ..., α_u and g are decided.
(7) The procedure is applied iteratively to α_1, ..., α_u and g.

2. Proper order of expansion

DEFINITION 4. Let X_c^i be a subset of a set of variables $X = (x_1, x_2, \ldots x_n)$. X_c^i has at most k elements. Let X' be a set such that $X' \cap X_c^i = \phi$, $X' \cup X_c^i = X$. For all the variables in X', give any constance to each variable. Then this set X' has r^{n-k} different states. Let each state be the element of a set B. Given an arbitrary function of n variables, $f(X) = f(X', X_c^i)$. $N^f(X_c^i)$ is defined as the number of the elements of a set B such that $f(B, X_c^i) \neq$ constance and $f(B, X_c^i) \neq x_j$, where $x_j \in X_c^i$. $N^f(X_c^i)$ is called 'dependency' of a function f(X) for X_c^i.

EXAMPLE 4. Consider the function of Table 2.
Let X_c^1 and X_c^2 be (x_1, x_2) and (x_3, x_4) respectivly.
Then by THEOREM's 1 and 2, $f((0, 0), X_c^2) = 1$,
$f((0, 1), X_c^2) = x_3$, $f((1, 0), X_c^2) = x_4$
and $f((1, 1), X_c^2) = 0$.
Therfore $N^f(X_c^2) = 0$.
Similarly $N^f(X_c^1) = 4$.

x_1, x_2 \ x_3, x_4	00	01	10	11
00	1	1	1	1
01	0	0	1	1
10	0	1	0	1
11	0	0	0	0

Table 2 The truth table of EXAMPLE 4

THEOREM 11. Let X be divided disjunctively to the number b of subsets $(X_c^1, \ldots, X_c^i, \ldots, X_c^b)$, where $b = \lfloor n/k \rfloor$. Let f(X) be expressed as

$$f(X) = U(f_0(X'), f_1(X'), \ldots, f_j(X'), \ldots, f_{r^k-1}(X') ; X_c^i).$$

In the total sum of 'dependency's', the following relation is hold for any i

$$\sum_{l=1}^{b} N^f(X_c^l) = N^f(X_c^i) + \sum_{m=0}^{r^k-1} \sum_{l=1}^{b}{}' N^f{}_m(X_c^l)$$

where Σ' denotes the summation except i.

PROOF. By THEOREM 5, $f_j(X') = f(X', X_c^i \leftrightarrow j)$. 'Dependency' of the function f(X) for X_c^l is then

$N^f(X_c^l) = N^{f(X', X_c^i \leftrightarrow 0)}(X_c^l) + \ldots\ldots + N^{f(X', X_c^i \leftrightarrow r^k-1)}(X_c^l)$ where $i \neq l$.

Therefore $N^f(X_c^l) = \sum_{m=0}^{r^k-1} N^f{}_m(X_c^l)$.

This implies that $\sum_{l=1}^{b}{}' N^f(X_c^l) = \sum_{m=0}^{r^k-1} \sum_{l=1}^{b}{}' N^f{}_m(X_c^l)$.

THEOREM 11 assures that if a subset $(X_c^1, X_c^2, \ldots, X_c^i, \ldots, X_c^b)$ is given, then the sum of 'dependency's' of each residue function and 'dependency' of the output function f(X) for the subset X_c^i is constant for any i.

THEOREM 12. Let a function of n variables be expessed in a tree struture, and let the number of the levels be b. In one U_l of the l-th level, 'dependency' of the value $r^{(b-l)k}$ can be admitted at its maximum.

PROOF. Consider a function of n variables ,where n = bk. For one set of k variables 'dependency' of the function is $r^{(b-1)k}$ at its maximum. By THEOREM 6, the number of U_l's is $r^{k(l-1)}$. So in one U_l, 'dependency' of the value $r^{(b-1)k}/r^{(l-1)k} = r^{(b-l)k}$ can be admitted.

THOREM 13. For k =1 and n =2, if $N^f(x_1) \geq N^f(x_2)$, it is the minimum realization that x_1 is assiged to the first level and x_2 is assigned to the second level (assume that identical residue functions can not be used as common inputs).

PROOF. The given function is expressed in two ways such as below.

$$f(x_1, x_2) = U(f_0(x_2), \ldots, f_{r-1}(x_2) ; x_1) \qquad (3)$$

$$f(x_1, x_2) = U(f'_0(x_1), \ldots, f'_{r-1}(x_1) ; x_2) \qquad (4)$$

In Eq.(3) the number of T-ULM's is $1 + N^f(x_2)$. In Eq.(4) the number of T-ULM's is $1 + N^f(x_1)$. Therefore the theorem is obvious.

EXAMPLE 5. Consider the synthesis of the ternary function of Table 3 using T-ULM's of one control variable. Since $N^f(x_1) = 3$ and $N^f(x_2) = 0$, x_1 is assigned to the first level. As a result $f(x_1, x_2) = U(x_2, 2, 1 ; x_1)$. If x_2 is assigned to the first level, then $f(x_1, x_2) = U(U(0, 2, 1 ; x_1), U(1, 2, 1 ; x_1), U(2, 2, 1 ; x_1) ; x_2)$.

x_1 \ x_2	0	1	2
0	0	1	2
1	2	2	2
2	1	1	1

Table 3 The truth table of EXAMPLE 5

We consider first the T-ULM of one control variable(i.e. k =1). The optimal solution is the assignment in which the number of T-ULM's necessary to realize the function becomes minimum. This problem is

attributed how to find the most imcomplete tree structure. By THEOREM 11, if to the first levle the variable for which 'dependency' of the given function is maximum is assigned, the total sum of 'dependency's' for the other variables becomes minimum. By THEOREM 12, the smaller l is, the larger value of 'dependency' can be admitted in one U_l of the l-th level. Therefore it is considered that the number of T-ULM's necessary to realize succesive levels may be smaller. Similarly at the next level, this procedure is continued. As a result at the last level the total sum of 'dependency's' becomes fairly small, and we can obtain a suboptimal solution.

We consider next the T-ULM of k control variables($k \geq 2$). This problem is not simple as the case of $k = 1$ because all the enumerations of the k control variables in each level are not practically possible for the function of large variables. However, we can say that to the upper level assigning the variables for which 'dependency' is small gives a preferable solution. As an extention for the case of $k = 1$, we select the k variables to be assigned to the first level for which 'dependency' is maximum. Then, 'dependency' of the function for the other variables may be smaller. Thus this procedure is carried out at each level. The above mentioned procedure is presented as Subalgorithm 2.

Subalgorithm 2.

(1) Pick up k variables from given n variables. The number of possible combinations is ${}_nC_k$ (if $n < k$, all the variables are picked up).
(2) For one of these combinations, 'dependency' of the given function f(X) is calculated.
(3) After all the combinations are checked, the variables X_c^i for which 'dependency' becomes maximum are selected. This X_c^i is assigned to the first level of the network of T-ULM's.
(4) Each residue function is calculated as $f_j(X') = f(X', X_c^i \longleftrightarrow j)$.
(5) The procedure is applied to each residue function up to the b-th level.
(6) By THEOREM's 1 and 2, the reduction of the number of T-ULM's is carried out.

IV. EXAMPLE

We shall consider a problem of synthesizing the ternary function of Table 4 using T-ULM's of one control variable. After all the combinations of the subsets Y and Z are checked, the pair of $Y = (x_1, x_3)$ and $Z = (x_2, x_4)$ is selected. Since $K(Y ; f) = 3$, $u = \lfloor \log_3 K(Y ; f) \rfloor = 1$. So, F of Eq.(2) is calculated as

$$F = (3^2 - 1)/2 + (3^3 - 1)/2 = 17.$$

As F is less than $(3^4 - 1)/2 = 40$, to the compatible sets the same code are assigned as shown in Table 5. The similar procedure is carried out in the function of $f = g_1(\alpha_1, x_2, x_4)$. As a result $f = g(\alpha_1, \alpha_2)$ and α_2 are given in Table 6. By Subalgorithm 2, α_1, α_2 and g are expressed as

$$f(x_1, x_2, x_3, x_4) = g(\alpha_1, \alpha_2)$$
$$= U(U(1, 0, 2 ; \alpha_1), U(0, 1, 1 ; \alpha_1), \alpha_1 ; \alpha_2).$$
$$\alpha_1 = U(x_3, U(1, 0, 1 ; x_3), U(2, 2, 0 ; x_3) ; x_1).$$

$\alpha_2 = U(0, U(1, 2, 0 ; x_2), x_2, ; x_4)$.

Thus the network of T-ULM's is realized as shown in Fig. 4.

x_1, x_2 \ x_3, x_4	00	01	02	10	11	12	20	21	22
00	1	0	1	0	1	0	2	1	2
01	1	0	0	0	1	1	2	2	1
02	1	1	0	0	0	1	2	2	2
10	0	1	0	1	0	1	0	1	0
11	0	1	1	1	0	0	0	1	1
12	0	0	1	1	1	0	0	0	1
20	2	1	2	2	1	2	1	0	1
21	2	2	1	2	2	1	1	0	0
22	2	2	2	2	2	2	1	1	0

Table 4 A ternary function of 4 variables

$\alpha_1(x_1, x_3)$

x_1 \ x_3	0	1	2
0	0	1	2
1	1	0	1
2	2	2	0

Table 5

$g(\alpha_1, \alpha_2)$

α_1 \ α_2	0	1	2
0	1	0	0
1	0	1	1
2	2	1	2

$\alpha_2(x_2, x_4)$

x_2 \ x_4	0	1	2
0	0	1	0
1	0	2	1
2	0	0	2

Table 6

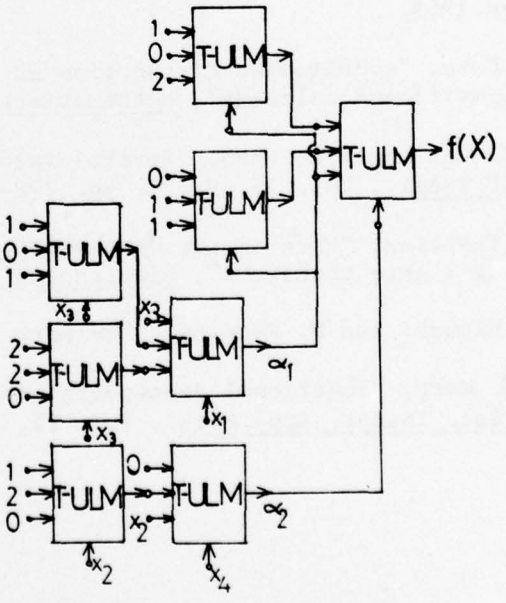


Fig. 4 The network of T-ULM's

V. CONCLUSION

In this paper, we have presented the mathematical properties of T-ULM in the multiple-valued logic. On the basis of these properties, an algorithm for synthesizing a network of T-ULM's with a smaller number of T-ULM's, has been discussed. The algorithm gives a suboptimal solution for the synthesis problem. Moreover, it is also possible to implement this procedure by a computer program. However, this algorithm does not always give the optimal solution. So, we need more sufficient studies as to the method for deciding the assignment $(\alpha_1, \ldots, \alpha_u)$, the method for using identical residue functions, and partially-specified functions.

ACKNOLEDGMENT

The authors are indebted to Prof. T. Anayama for his guidance and encouragement.

REFERENCES

[1] S.S. Yau and C.K. Tang, "Universal logic circuits and their modular realizations", 1968 Spring Joint Computer Conf., AFISP Proc., Vol. 32, Washington, D.C. : Thompson, pp. 297-305, 1968.

[2] S.S. Yau and C.K. Tang, "Universal logic modules and their applications", IEEE Trans. on Computers, Vol. C-19, pp. 141-149, Feb. 1970.

[3] E.L. Post, "Introduction to a general theory of elementary propositions", American Jounal of Mathematics, Vol. 43, pp. 163-185, 1921.

[4] D.I. Porat, "Three-valued digital system", Proc. IEE, Vol. 116, pp. 947-954, June 1969.

[5] A. Rose, "conditioned disjunction as a primitive connective for the m-valued propositional calculus", Mathematische Annalen, Vol. 123, pp. 76-78, 1951.

[6] C.Y. Lee and W.H. Chen, "Several-valued combinational switching circuit", AIEE Trans., Vol. 75, Pt. I, pp. 278-283, July 1956.

[7] S. Thelliez, "Note on the synthesis of ternary combinational networks using T gate operators", Electronics Letters, Vol. 3, pp.204-205, May 1967.

[8] T. Higuchi and M. Kameyama, "Ternary logic system based on T-gate" (this issue).

[9] R.M. Karp, "Functional decomposition and switching circuit design", J. Soc. Indust. Appl.Math., Vol. 11, pp. 291-334, June 1963.

ASSOCIATIVE MEMORIES AS MULTIPATH LOGIC SWITCHES

Yoh-Han Pao and Jeffrey Altman
Case Western Reserve University
Cleveland, Ohio U.S.A.

Recently Pao and Merat[1] reported on a new type of distributed associative memory which may be used for pattern recognition. In this paper we indicate how such a memory may be used as a multipath logic switch. Such switches may be implemented in "software" to provide multiple-valued branching in higher level language programming or it could be implemented in special purpose dedicated hardware devices. This type of switch is interesting in that the incoming and outgoing path relationships vary depending on the "context" condition. Applications in computer aided instruction, fault location systems and machine translation are clearly indicated, but remain to be implemented.

Initially, we will review some characteristics of the Pao and Merat distributed associative memory.

Let such a memory contain N storage sites and let the stored value at each site be either +1 or -1. In the present context, a pattern is a sequence of +1 and -1 values.

In the detailed computer simulated study reported by Pao and Merat, the memory contained 64 storage sites and was suitable for storage of patterns consisting of sequences of sixty four values of either +1 or -1.

For pictorial purposes, the storage sites might be represented by squares and might be colored black or white depending on whether +1 or -1 is represented. Furthermore for pictorial purposes, the sixty four squares may be arranged in the form of an eight by eight array of squares. However, the patterns can be represented equally well by a linear array of colored squares.

We note that even a sixty-four space is not trivial, nor is it of limited interest. An eight by eight display such as that shown in Figure 1 can in fact be used to display any one of 2^{64} different patterns, that is, any one of 18446744074709551616 different patterns.

Ordinarily, if a pattern is stored in such a space then that memory storage space has been used up and further space is needed to sine a second pattern. Furthermore, if the total number of such patterns to be stored is P, then the total number of storage sites required is NP. If a duplicate of any one of the stored patterns is presented for recognition, a template matching technique would involve the comparison of the incoming pattern with most of stored patterns and the number of basic processing steps required would be on the order of KNP where K is an integer on the order of $1 \sim 10$.

In contrast, the Pao and Merat method requires a storage space of (m+1)P, where $2^m = N$, and the overall processing time is correspondingly decreased from KNP to K'(m+1)P where K' is not necessarily equal to K but is also of the order of $1 \sim 10$.

The basis for the operation of this distributed associative memory can be made evident by the following brief theoretical discussion.

Let each pattern, that is each sequence of +1 and -1 values be represented by a vector. That is, in N space, let

$\underline{x} = (x_1,\ldots,x_N)$ and $\underline{y} = (y_1,\ldots,y_N)$ be two N-component vectors.

We now define a vector product of the two vectors by

$$\underline{z} = (z_1,\ldots,z_N) = (x_1y_1,\ x_2y_2,\ \ldots\ ,\ x_iy_i,\ \ldots\ ,\ x_Ny_N) \tag{1}$$

That is, the ith component of the product vector is the arithmetic product of the ith component of the factors.

We note that this is associative so that $\underline{u}(\underline{v}\ \underline{w}) = (\underline{u}\ \underline{v})\underline{w}$

We also define a scalar product $\langle\underline{z}\rangle = \langle\underline{x}\ \underline{y}\rangle$ by

$$\langle\underline{z}\rangle = \langle\underline{x}\ \underline{y}\rangle = \sum_{i=1}^{N} x_iy_i = \sum_{i=1}^{N} z_i \tag{2}$$

In particular, since the elements of vector $\underline{x}$ are either +1 or -1, it follows that

$$\underline{x}\ \underline{x} = (1,1,1,1,1,1,\ \ldots\ ,\ 1,\ \ldots\ 1), \text{ the vector of all ones} \tag{3}$$

and
$$\langle\underline{x}\ \underline{x}\rangle = N\ . \tag{4}$$

In N space, there are 2^N different vectors which represent the 2^N different patterns each of which can be displayed using an array of N squares or sites. Of these 2^N different vectors (patterns), there are N vectors (patterns) which are distinct from the rest and form one set of orthonormal basis vectors which span this space. These latter patterns are called references (patterns). That is, if P patterns are of interest, then these may be represented by $\underline{x}_\alpha$, $\alpha = 1,2,\ \ldots\ ,\ P$ and the references are represented by $\underline{y}_\alpha$, such that

$$\langle\underline{y}_\alpha\rangle = \sum_{j=1}^{N} (y_j)_\alpha = 0 \qquad \text{for all } \alpha \tag{5}$$

and
$$\frac{1}{N}\ \langle\underline{y}_\alpha\underline{y}_\beta\rangle = 1 \qquad \text{for } \alpha = \beta \tag{6}$$
$$= 0 \qquad \text{for } \alpha \neq \beta$$

where the lower case Greek alphabet subscripts α (and β) are used to indicate the identity of the reference patterns.

Vectors with such properties do exist and are related to Walsh functions of different sequences.[2] Such vectors also have the group property that

$\underline{y}_\alpha\underline{y}_\beta = \underline{y}_\gamma$ when $\underline{y}_j$ is also a member of this same subgroup of patterns and

$\gamma = \alpha \oplus \beta$ where $\oplus$ indicates addition modulo 2.

In this scheme, a reference $\underline{y}$ is associated with each pattern $\underline{x}$ and the memory vector $\underline{m}$ is obtained by forming the vector product of the pattern and the associated reference, that is

$$\underline{m} = \underline{x}\,\underline{y} \tag{7}$$

Then, given a pattern $\underline{x}$ and the memory $\underline{m}$, one forms

$$\underline{x}\,\underline{m} = \underline{x}(\underline{x}\,\underline{y}) = (\underline{x}\,\underline{x})\underline{y} = \underline{y} \tag{8}$$

to retrieve the reference $\underline{y}$.

If we can unmistakeably (and with little effort) recognize that $\underline{y}$ has been retrieved, then pattern $\underline{x}$ has in fact been recognized.

One of the ideas underlying this present technique is to store many such individual memory vectors $\underline{m}_\gamma$ in the same storage space without regard for the fact that the space had already been used in a certain sense by other $\underline{m}$ vectors.

That is, when one has P patterns $\underline{x}_\alpha$ ($\alpha = 1, \ldots, P$) and associated references $\underline{y}_\alpha$, (the $\underline{x}$ and $\underline{y}$ vectors are associated in a specific and possibly trainable way), the individual memories are

$$\underline{m}_\alpha = \underline{x}_\alpha \underline{y}_\alpha \tag{9}$$

and the overall associative memory is

$$\underline{m} = \sum_{\alpha=1}^{P} \underline{x}_\alpha \underline{y}_\alpha \tag{10}$$

where the notation used in (10) indicates that the jth component of $\underline{m}$ is the sum of the jth components of all the $\underline{x}_\alpha \underline{y}_\alpha$ vectors.

Of course, now $\underline{m}$ is no longer binary, it is multivalued and retrieval is no longer so clear. In particular, if $\underline{x}_\eta$ is to be recognized, we carry out the following operation

$$\underline{x}_\eta \quad \underline{m} = \underline{y}_\eta + \sum_{\alpha \neq \eta} \underline{x}_\eta \underline{x}_\alpha \underline{y}_\alpha \equiv \underline{r}_\eta \tag{11}$$

where $\underline{r}_\eta$ is in general $\underline{y}_\eta$ with additional contributions from other $\underline{y}$ vectors.

A discriminant vector $\underline{f}$ is then chosen to emphasize a certain context or certain "features" or provide greatest resolution in certain regions of the N dimensional pattern space. In general, $\underline{f}$ is a linear combination of the references $\underline{y}_\beta$, that is

$$\underline{f} = \sum a_\alpha \underline{y}_\beta \tag{12}$$

where a_β are expansion coefficients.

We form an identifier D_η by the scalar product operation

$$\langle \underline{r}\, \underline{f} \rangle = a_\eta + \sum_{\eta \neq \alpha} \sum_{\beta} a_\beta \langle \underline{x}_\eta \underline{x}_\alpha \underline{y}_\alpha \underline{y}_\beta \rangle = D_\eta \quad \text{(a real number)} \tag{13}$$

In order to be able to recognize that $\underline{x}_\eta$ was presented for recognition, it is only necessary that the association

$$\underline{x}_\eta \iff D_\eta \tag{14}$$

be unique and that the difference between D_η and any other D be measurable with sufficient resolution.

To appreciate the merits and disadvantages of this type of memory, we compare this technique with that of straightforward template matching. In particular, consider $P = 10^6$ patterns each with $N = 10^6$ elements. In the template matching technique, $NP = 10^{12}$ binary storage sites, and about 10^{12} processing steps are required for determining which patterns had been presented for recognition. These overwhelming requirements in one form or another constitute the very real limitations to the recognition of complex patterns in real time.

In contrast to that, the distributed associative memory technique required only 21×10^6 binary storage sites ($2^{20} > 10^6$, and one additional bit is used to denote the sign of the stored value) and only 21×10^6 processing steps are required, representing five orders of magnitude decrease in storage capacity and about four orders of magnitude decrease in processing. It is clear then that even if each basic processing time interval lasted 10^{-6} seconds, it is clear that recognition of a pattern would require 10^6 seconds or approximately 278 hours, whereas the same task can be achieved in 21 seconds using the distributed associative memory.

Pao and Merat investigated a sixty-four storage space for different numbers of patterns up to 44 patterns and found no difficulty in storage or recognition.

The reference patterns are illustrated in Figure 2 and the storage and retrieval of associated references are exhibited in Figures 3,4, and 5, at first only one associated pair and subsequently for a large number of patterns. In these figures the multiplication sign (x) denotes vector product and it also signifies taking the arithmetic product of each pair of components. In Figure 5, the operations represented by equation (10) is represented graphically. The operations represented by equation (11) is shown schematically in the lower part of that same figure.

In their study, Pao and Merat investigated the performance of this type of memory for forty-four stored patterns and found no ambiguity in recognition.

We now proceed to show how this type of memory may be used as a multipath switch. The system flow chart is illustrated schematically in Figure 6, where we indicate that in some computation, the result of the computation is a list of sequence of binary values (+1 or -1 in our discussion but readily converted to 1 or 0 in a computer calculation). We now use an associated memory to implement a computed GO TO instruction.

In the present discussion, we use sixty-four randomly generated patterns. These can be exhibited in the form of 8 x 8 spatial arrays as shown in Figure 7 or as linear sequences as exhibited in Figure 8. Incidentally, using the same reference as those exhibited in Figure 2, the associative memory is calculated to be the array of sixty-four numbers also exhibited in Figure 8.

In Figure 8, pattern number 34 is also exhibited separately on the left side of the complete array of the sixty-four patterns. Typically, it takes about 20 seconds for a human to recognize that pattern as the third pattern in the fifth column. Using the associative memory scheme, it is estimated that the same task can be accomplished in less than a millisecond, using a storage space of 384 binary storage sites instead of 4096 sites.

For different discriminants $\underline{f}_\zeta$, we have different $\underline{x}_\eta$ <=> $D_{\eta\zeta}$ associations. Therefore given the "context" f_ζ, input $\underline{x}_\eta$ yields a computed value $D_{\eta\zeta}$ which with the aid of a look-up list yields the next instruction $I_{\eta\zeta}$. However if the "context" is $f_{\zeta'}$ (with $f_{\zeta'} \neq f_\zeta$), then the same $\underline{x}_\eta$ would be assigned a different path via $D_{\eta\zeta'}$ to $I_{\eta\zeta'}$.

In principle, the quantities $D_{\eta\zeta}$ can be specified for all $\underline{x}_\eta$ and the expansion coefficients $a_{\zeta\beta}$ in $f_\zeta = \Sigma\, a_{\zeta\beta} y_\beta$ by solving the set of simultaneous living equations

$$\langle \underline{x}_\eta \underline{m} \underline{f}_\zeta \rangle = D_{\eta\zeta} = \sum_\beta a_{\zeta\beta} \langle \underline{x}_\eta \underline{m} y_\beta \rangle \qquad (15)$$

In practice, it is more convenient to use any one of the reference patterns $\underline{y}_\beta$ as the discriminant. The advantage is that $\underline{y}_\beta$ is easily generated and there is no need to store an additional set of $a_{\zeta\beta}$ coefficients. The disadvantage is that such arbitrarily chosen discriminants do not necessarily yield a set of $D_{\eta\zeta}$ values which are all distinct from each other. A simple remedy for this is to use a second discriminant to distinguish further in those cases where the first discriminant cannot yield a unique decision. As an example, the $D_{\eta\zeta}$ values for $\eta = 0, \ldots, 63$ and $\zeta = 0,1, \ldots, 10$ are shown in Table 1. In operation as a multipath switch, if $\underline{X}_6$ is the input and if $\underline{f} = \underline{y}_0$ is the "context" (or discriminant), the D_{60} is computed to be 26 and reference to a look-up list yields the address for the next instruction. However, if D is computed to be -54, the look-up list would yield the instruction that an additional discriminant (say $f = \underline{y}_1$) be used. If the result of this second calculation is -42, then the look-up list will yield the information that the incoming pattern was $\underline{X}_7$ and the next instruction is accordingly $I_{(-52,-42)}$, which of course may be anything which the programmer wishes to make it. Similarly if the results of the second calculation was 6, then it is known that the incoming signal was in fact $\underline{X}_{14}$.

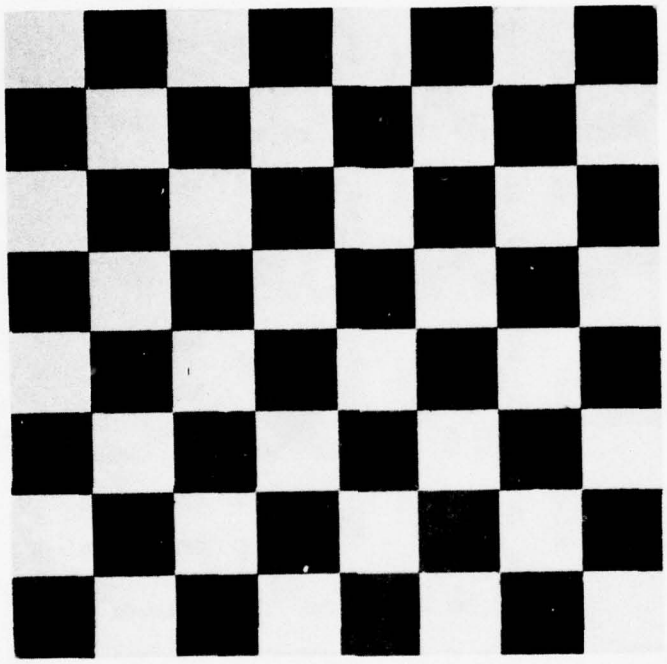
This type of multipath logic switch permits iterative convergence so that

the correct "context" might be searched for to yield the optimum result. It would seem that such multipath logic switches can find ready application as unit operations in a higher level language. Other applications are also indicated, some of these being monitoring of progress in computer monitoring of self-paced education, fault location in complex integrated circuit systems and machine translation of languages.

References

1. Yoh-Han Pao and Frank L. Merat, "Distributed Associative Memory for Patterns", Case Western Reserve University, Electrical Engineering and Applied Physics Internal Report, also submitted for publication to IEEE Transactions on Man, Systems and Cybernetics.

2. Henning F. Harmuth, "Transmission of Information by Orthogonal Functions", p. 20, Springer-Verlag, New York Inc., 1969.



2^{64} POSSIBLE PATTERNS

$2^{64} = 18446744073709551616$

Figure 1. Number of Different Patterns Which May be Displayed Using an 8 x 8 Array of Picture Elements (Each Picture Element Being Either +1 or -1).

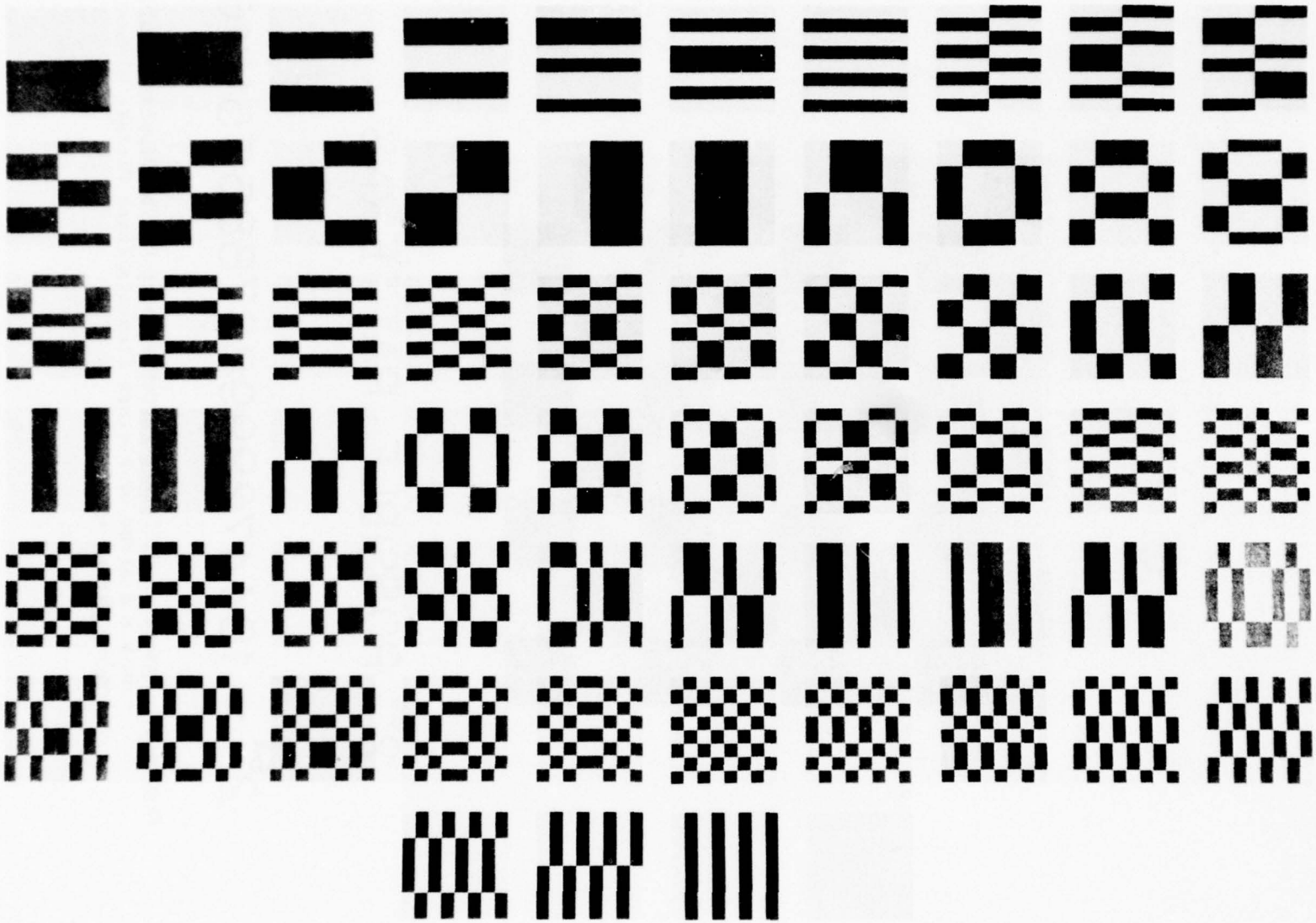


Figure 2. Orthonormal Reference Vectors (Walsh Functions) For a 64-Element Pattern Space. Sequency Increases From Left to Right In Each Row and from Row to Row Except That the Last Pattern is Wal(0,t).

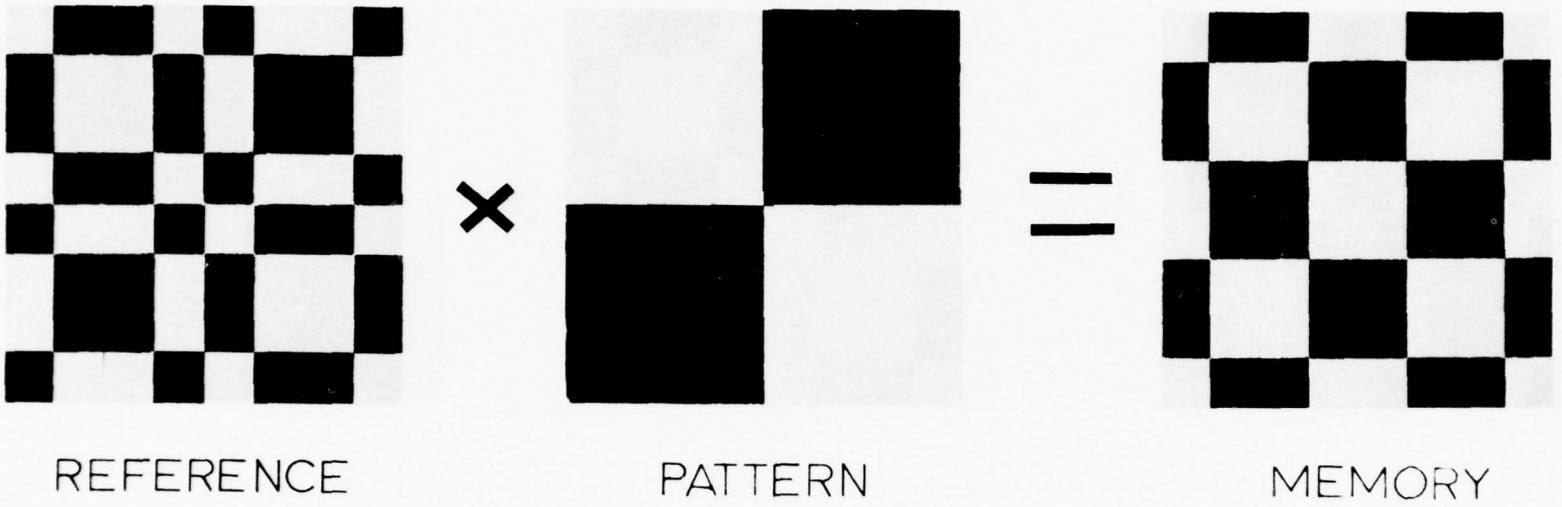


Figure 3. The Recording Process.

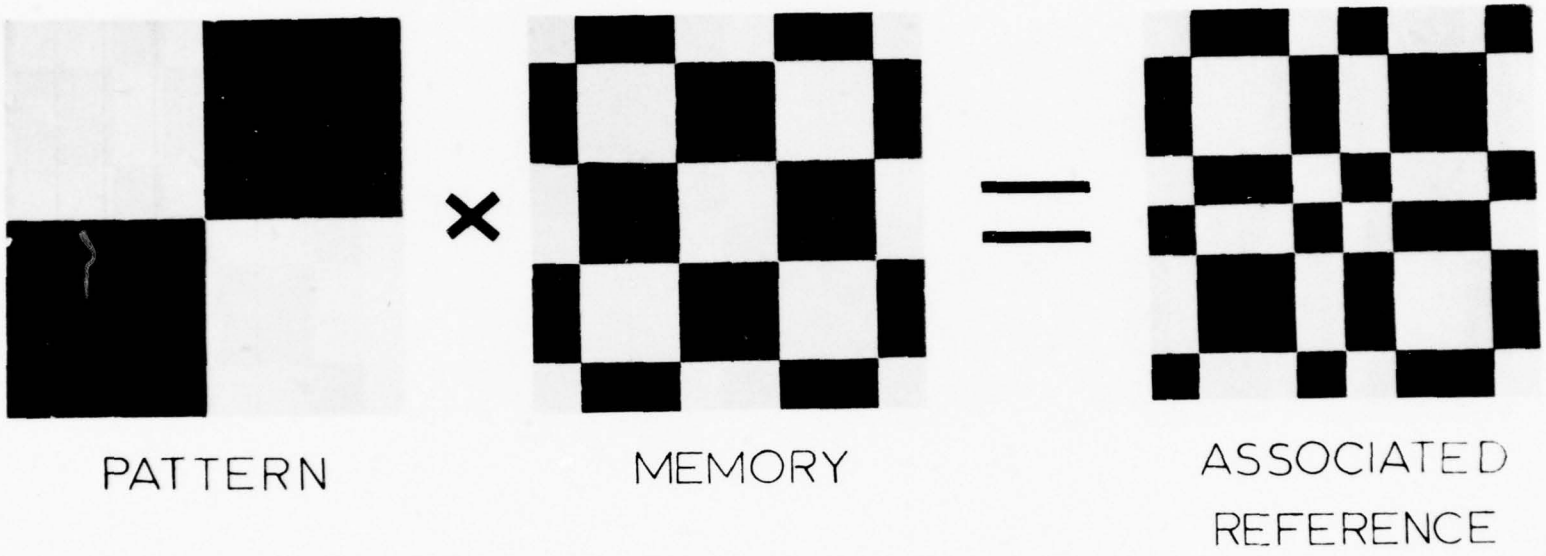


Figure 4. Regeneration of the Associated Reference.

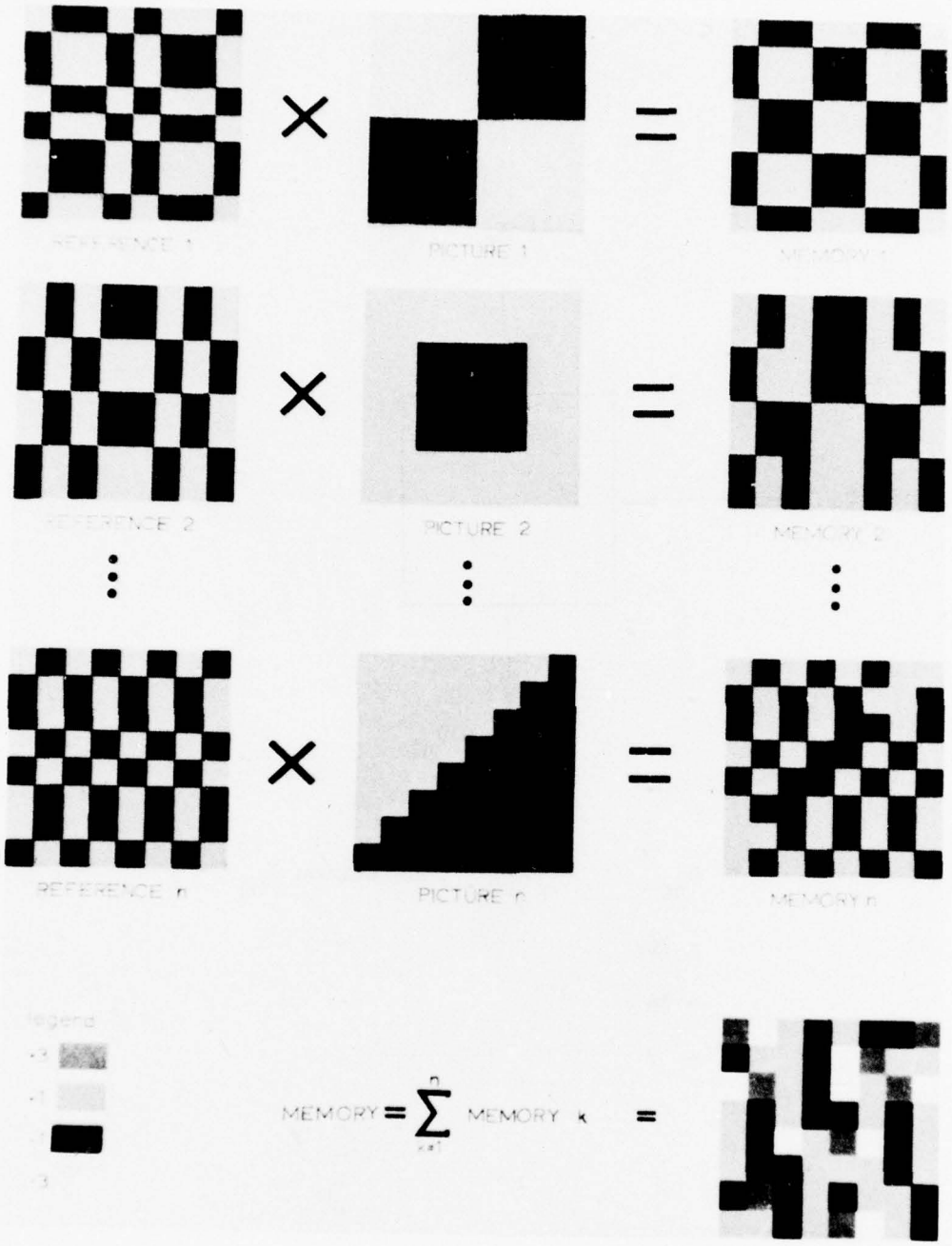
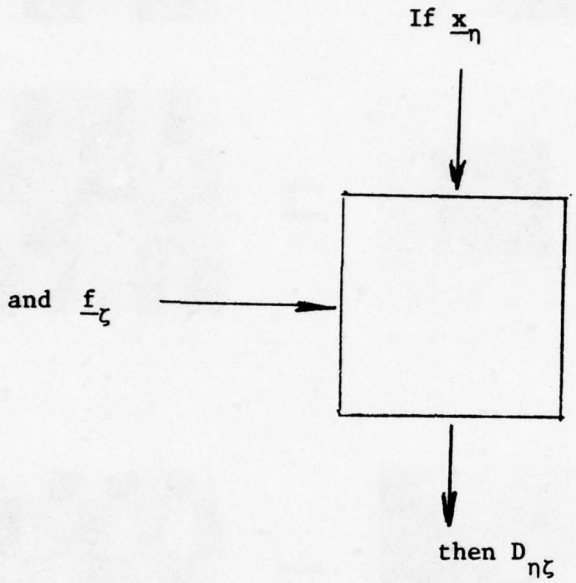


Figure 5. Construction of a Memory Containing Many Associated Pairs of Patterns and References.



$D_{\eta\zeta}$	Instruction $I_{\eta\zeta}$
- 26	-
- 14	-
- 2	-
.	-
.	-
.	-
.	-
.	-
.	-
118	-

Figure 6. Schematic Illustration of Multipath Logic Switch

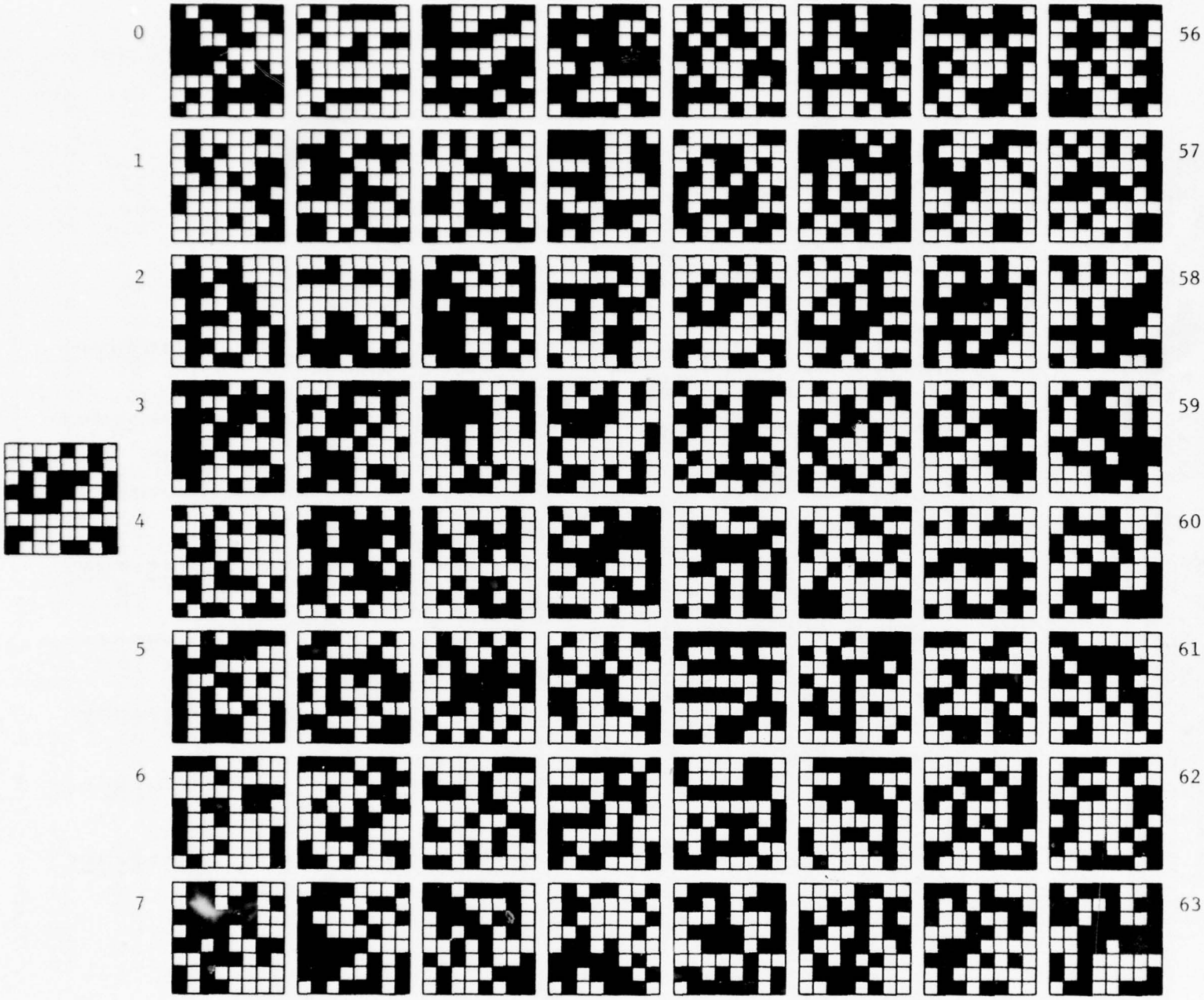
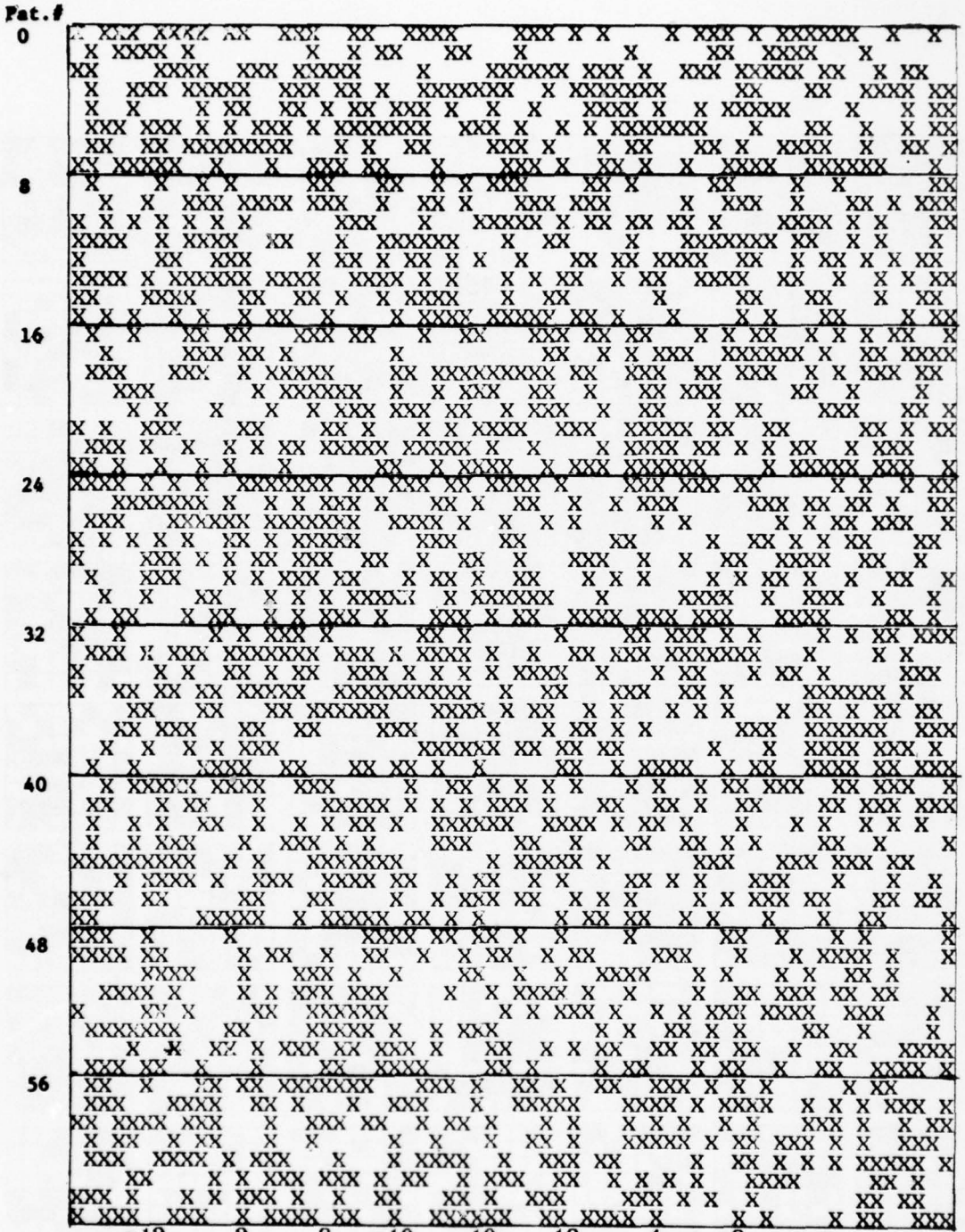


Figure 7. Sixty-four Randomly-Generated Patterns Exhibited As 8 x 8 Arrays.



-12	2	8	-10	10	18	4	-8
4	8	-10	4	-10	-4	-4	0
0	2	-4	-2	14	4	4	-2
-10	10	6	8	6	8	-10	-14
-6	-2	-2	-10	0	0	20	-2
-2	12	-2	4	-12	-4	8	-6
-10	0	-4	0	10	4	4	6
4	-4	2	-4	0	-2	-8	-12

Figure 8. Sixty Four Randomly Generated Patterns Exhibited As Linear Sequences. Also Equivalent Associative Memory.

DISCRIMINANTS

F=WAL(0)

118	118	-34	114	-10	6	26	-54
30	-6	38	-74	-62	-6	-54	-82
98	-82	-78	90	58	-90	166	34
-110	86	-54	42	94	58	34	90
-126	70	-26	-46	82	-70	82	-6
-46	-10	6	10	70	-34	6	22
-34	-54	66	-10	-46	86	-22	-2
86	-78	-14	-34	-6	-14	-54	-58

F=WAL(1)

-30	82	-102	94	66	122	-74	-42
-46	94	-62	10	30	30	6	10
94	18	-42	-2	-34	34	-14	-98
38	114	-26	14	10	70	-2	-58
-26	66	-6	-42	94	-58	-2	62
38	34	58	54	-22	-22	2	-22
106	-18	22	58	46	34	-98	-46
106	62	86	-70	30	-34	-34	26

F=WAL(2)

-34	46	6	-78	-26	30	42	18
54	-30	-18	70	-70	-54	18	-18
-54	-2	-14	58	-70	-2	-42	18
-22	-34	18	2	-26	-54	90	-54
-94	-26	14	-6	34	-38	-30	10
106	14	38	26	86	-18	14	-138
62	42	-6	54	74	-42	-30	70
-58	-54	42	-2	-30	66	2	-98

F=WAL(3)

-70	-22	-30	110	82	-14	166	-2
106	22	-6	-54	-154	-50	-18	26
6	34	54	-34	46	26	66	14
-2	42	-66	-26	-30	54	38	-10
-122	-46	-94	-194	-34	54	-34	-34
30	138	-22	-74	-6	-6	26	-38
26	-2	-18	-70	-26	66	86	186
-38	-42	-34	10	38	14	-42	18

F=WAL(4)

-66	14	-66	58	70	-82	2	34
-58	154	-50	6	-22	-94	10	-10
10	70	-62	34	2	6	46	-62
-86	-58	-14	-14	70	-14	34	-30
26	62	-58	-22	50	-70	130	-38
98	-74	-2	50	-58	-42	134	6
-2	10	58	-26	-30	102	-70	22
82	-70	34	-58	18	-14	50	-26

F=WAL(5)

-38	26	-86	54	-110	34	46	-34
-38	78	42	-70	-10	22	6	50
22	-6	6	38	22	82	-38	-98
-2	82	-50	54	-30	46	-66	30
62	-22	-102	-34	46	6	158	46
54	-78	34	46	-38	34	34	-22
10	14	14	42	-66	-14	-82	-54
42	82	38	18	-10	78	70	-70

F=WAL(6)

-98	30	30	18	-18	14	90	34
-90	-70	46	-34	-70	42	42	-18
-70	-66	10	26	90	38	38	-150
58	-26	-46	18	-106	90	50	-38
-94	38	-26	10	42	34	26	18
66	6	54	-54	14	78	-42	-34
22	2	10	94	82	-2	42	6
78	10	-110	-18	-126	202	82	-58

F=WAL(7)

-6	-6	-38	78	42	18	118	14
42	78	74	-78	6	-18	-42	-86
-10	66	-98	-18	14	-30	-30	-10
-2	-78	14	-26	-46	-90	142	6
-74	-78	74	-18	38	-18	-10	38
70	98	-6	-26	50	-22	50	-30
-30	70	-18	50	30	58	110	74
-46	-74	54	-38	-74	54	-42	-22

F=WAL(8)

-58	78	-138	-86	-34	-2	-46	-118
30	-38	-106	-18	66	-94	2	-26
-14	22	82	26	2	-42	-42	-62
-86	-34	74	-6	-42	90	-14	-22
2	-34	-42	-62	18	18	34	18
50	-10	14	42	-50	46	38	-2
70	-54	66	-58	-54	14	18	-50
-2	74	-46	70	2	-14	98	-58

F=WAL(9)

-30	-102	2	-90	-22	18	-2	54
18	62	50	2	30	-58	-82	-46
30	170	6	-50	54	-46	-46	62
-66	-86	-26	-162	50	6	30	-10
54	-102	26	-74	-82	30	14	102
38	18	-30	54	-62	-22	18	2
2	-82	38	-38	-74	-38	54	2
-14	118	-106	50	6	-66	-42	-38

Table 1. Identifiers $D_{\eta\zeta}$ For Different Patterns $\underline{X}_{\eta}$ and Different Discriminants $\underline{f}_{\zeta}$.

ASSOCIATIVE AND MULTI-VALUED LOGIC FOR POSSIBLE IMPROVEMENTS IN SOME X-RAY IMAGE PROCESSING

David Rine
Computer Science
West Virginia University

There are at least three functions for the design of digital image processing systems: (1) the rate at which digitized imagery data must flow through the system, determined by the nature of the digital encoding of the image; (2) the complexity of the image transformation; (3) the type of output required by the user. A digitized image is broken down into frames, and each frame is broken down into the basic primitive picture cells or pixels; encoding begins at the pixel level, be it binary, octal, hexadecimal, or whatever. One of the fundamental operations performed per pixel is the simple subtraction of images. The digital image systems division of Control Data Corporation has used image subtraction quite recently [9] in the analysis of aerial photo scenes and in the search of malignant lesions in chest X-ray images. Multi-processor performance is related to input data rate (pixels/sec.), number of channels, operations/pixel, and operations/sec.

Multiple-valued logic has the potential for simplifying the basic encoding of image data.

The application area of this paper concerns the ranking of a class of X-ray images denoting progression of a lung disease by enlarging densities.

First, the subtraction operation has the potential of extending to the ranking procedure. So, with the promise that multiple-valued logic will simplify encoding of image data, the paper describes multiple-valued logic image subtraction techniques. This would enhance techniques already being used by CDC.

Second, since ranked or ordered output of images is ultimately needed, and since the differencing technique is slow when performed over many images and leads to ambiguous results when more than a detected difference is desired, an associative processing technique is clearly called for. Thus, several methods for properly extending associative processing techniques to image frames stored in many square array words are considered. Search paths which associatively operate through all frames or array words simultaneously to output or select them in order, parallel by plane and serially through every cell of every plane, are described. This method can be viewed as a generalization and modification of an algorithm originally introduced by Seeber and Lindquist [13]. The design of this second and more promising method, from a ranking point of view, is done with multiple-valued logic with its promise of simplified encoding of data.

Also, from a completely different point of view, with the possibilities of multiple-valued logic one might consider quite a variance in the selection of logic used in interrogating the associative memory. Positive reasons for exploring this include (1) further possible reduction in program complexities, (2) more efficient potential storage and encoding of data, (3) possible reductions in wiring complexity. Negative reasons, though, include the possibility that serial response time could be slower against words of associative memory, unless the shorter possible word length (digits per plane) offsets any slow serial operation times.

Improvements in image processing throughput will be directly related to the number of digits (binary or otherwise) that can be processed simultaneously. Associative/parallel processing computers have been considered as products of computer architecture for improving throughput, assuming that corresponding improvements in peripherals and communications devices can also be made. For some associative/parallel computers program complexity of, for example, a pattern matching routine is related to the (square

or linear) word length (number of bits or digits per word) when using serial by bit logic, and one must discover ways of reducing word length while preserving at least the same amount of image information. One way of improving serial operations might be to use multi-valued logic memories, if serial speed were to be gained by means of shortened word length. If a pattern or pattern window uses 256 bits of two-state memory, then it would use approximately 85 digits of 8-valued logic; also, proportionately fewer square words (planes) would be used in the encoding of image data.

A conventional computer has a location-addressed memory. For each input item, it must search all contents of its data file one at a time until it finds the data it needs, or engage in complex and time consuming software routines for storage or retrieval. Although file structures and programming techniques have lessened retrieval problems in the conventional processor, these techniques lead to other disadvantages in cost and program execution time. For each input data item, such as a part of a digitized image, STARAN-S can search all contents of its data file and identify in a single memory access all elements that meet the search criteria (e.g., an X-ray density description, such as progressive massive fibrosis, of a given ranking criterion).

Let us suppose that we are looking at a 32 x 32 grid with (as a special case of grey coding) black or white colored squares (see Fig. 1.). This grid image might represent

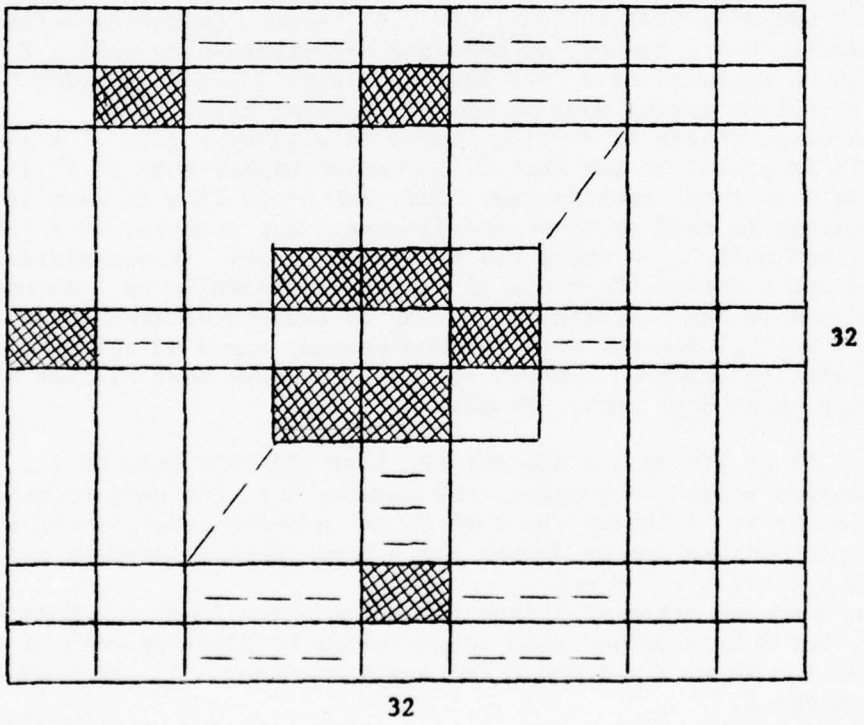


FIGURE 1.

a black and white X-ray photo of a patient's lung where the black on the above grid might, for convenience, be the predominantly white colored areas of the X-ray photo. While radiologists look, as through a window, at only a portion or region of the X-ray photo at a time, an interesting feature of the Iterative Array of Illiac III is that it may be thought of as a "window" viewing a 32 x 32 bit portion of the picture and of height equal to the number of planes necessary to represent the grey level encoding; and, an important feature is that the number of planes could very well be reduced by multiple-valued logic encoding and logic design.

One wishes to search the binary encoded digitized set of X-ray photos for the presence or absence of "information" at a particular coordinate, selecting in order those that have the "information" on which one is keying.

For picture data, a problem becomes one of selecting the "best" starting position, if any, and determining how to progress through the 32 x 32 array word of 1024 bits, for example. It is well-known that there are two "topologies" for the Illiac III arrays (planes, square words), rectangular and hexagonal. A triangle topology is, also, possible but is not considered on Illiac III.

Suppose now that in order to select the picture data in the right order (e.g., more dense than, reflecting true growth rate of a progressive massive fibrosis density) one needs to develop a "path" that goes through every cell beginning with a starting cell indicated by a dot "0" and following the direction rule that "1" denotes go to right neighbor cell, "2" bottom neighbor, "3" left, and "4" top neighbor cell. Thus, a path as indicated beginning with 0 and going by means of a "simple path (no intersecting or looping) strategy" through every cell is codified:

STRATEGY 1:

1 2 33 44 111 222 3333 4444 11111 22222 333333 ...

The 1's and 2's follow a repetitious pattern of 1,3,5, etc., while the 3's and 4's follow a repetitious pattern of 2,4,6, etc.

If one uses a 0 to represent a blank and a 1 to represent a solid cell, then A is 0101111111111110...0 and B is 01010101101111110...0. Hence, following strategy 1 the Boolean lattice ordering $0 \leq 1$ induces an ordering coordinate-wise making $B \leq A$. This Boolean ordering can be replaced by a Post lattice linear (chain) ordering $0 \leq 1 \leq 2 \leq \ldots \leq n-1$ of multi-valued logic, Boolean being a special case.

In the Boolean case, if one is sorting images in a given region of a given set of coordinates, then it is useful to say that "A is ranked higher than B" if it has more 1's, is denser, than B at those coordinates. And, one would like to know if it makes a difference what topology is used at those coordinates. For example, if $B \leq A$ under the rectangular array, then will $B \leq A$ under the hexagonal array? If one places the hexagonal topology on A and codifies the sides of a hexagon clockwise by 1 through 6, then the previous closed simple path strategy of moving by successive sides from cell to cell, in this case 1 2 3 4 5 ..., does not work for honeycombs, since in doing so a cell (next to the start cell) will be omitted. Hence, one uses the next most similar simple path codification strategy which does work. Namely,

STRATEGY 2:

1 3 4 5 6 11 2 33 44 55 66 111 22 333 444 555 666 1111 222 3333 4444 55 ... .

This path technique would not complete the square; but, the pattern underlying this code is that the 2's lag by -1 behind the rest of the members 1,3,4,5,6 of the sequence.

The following proposition can be proved for a Post lattice ordering replacing $0 \leq 1$ with multi-valued $0 \leq 1 \leq 2 \leq \ldots \leq n-1$.

Proposition. Given a square array digitized black and white image A and any starting point in the array, let B be a second such image formed by deleting zero or more 1's (black squares) from A. Then, $B \leq A$ under rectangle strategy 1 if and only if $B \leq A$ under hexagonal strategy 2.

Applicably speaking, the multi-valued logic can be viewed as being similar to the integer coding in a single plane for grey coding as it appears in software in the PAX II Emulator for Illiac III.

It can be seen that the implementation of an associative type memory unit by use of a ternary interrogation counter similar to, but more general than, that of Seeber and Lindquist [13] is entirely possible. The memory unit which one would choose, along with its being associative, would use a parallel-by-bit (STARAN is serial-by-bit) and parallel-by-plane (array) approach. The rules for the operation of the ternary counter are, as indicated [13], a function of the match indicator MI. Each setting of the ternary counter, except for the final all-M, M a mask bit, setting, constitutes at least one "interrogation" or "retrieval" try. Some of the steps in the retrieval tries could be avoided

if one knew in advance how many images (binary or multiply digitized on a plane or array) were in our sort, but the more general case is being considered where the size of the group (set) of image data is unknown.

Insofar as modifications to the retrieval algorithm, the generalization of the simple one in Seeber and Lindquist [13], are concerned one could have a more general match indicator circuit giving indications 0,1,2,P (and using multi-stable states for the match indicator) where P is now greater than two, more than two matches; one can save an occasional interrogation by designing the previously called "ternary counter" to "remember" when the first of a series of MI's equal to 2 are indicated.

The usual additional functional features where the complement of M,0,1 is defined to be M,1,0 are to included.

x	C(x)
M	M
0	1
1	0

It is obvious (using exclusive-OR with match indication) that in order to perform a descending sort, starting from a given image, the image must be stored in the ternary counter in its complement form, $M \rightarrow M$, $0 \rightarrow 1$, and $1 \rightarrow 0$.

Let, for example, the simple picture data

0	1
1	0

With the ternary counter the operating position OP is defined as the lowest-ordered position along the specified simple path (beginning at fixed coordinates) whose bit state is to be compared with every square word in the associative memory array store; and, if the match indication uses exclusive-OR logic -- emitting success responses to the response store RS (see Fig. 2.) then the picture data placed in the square word comparand register CR must be

1	0
0	1

.

The multi-valued associative-array search logic appears later in the paper.

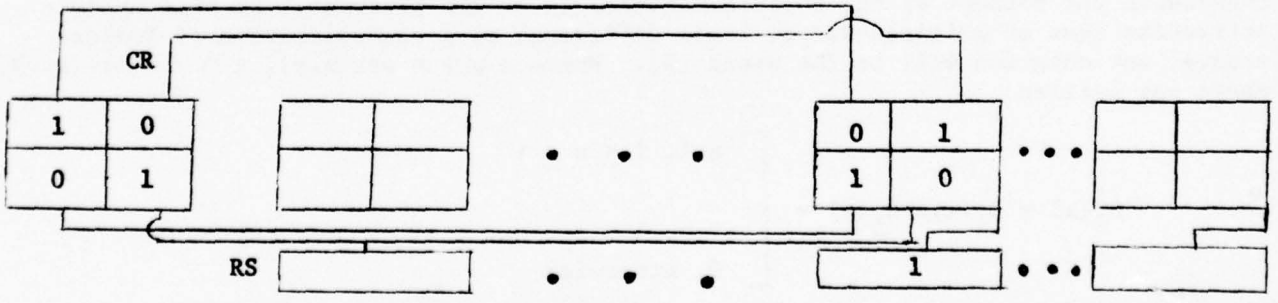


FIGURE 2. Array and RS with 2 x 2 Planes.

In the previous part of the paper one was given an ordering $B \leq A$ of images under a simple path strategy. And, one said that as long as 1 bits are being deleted that $B \leq A$ will be the case for any simple path strategy. Some reasoning needs to be given why this ordering or ranking of images is important, as opposed to the standard techni-

que of subtracting images.

First, let us review the well-known mask logical operation, exclusvie-OR, denoted by EXOR and defined by bit by the following table. Recall that the exclusive-OR

	X_2	
EXOR	0	1
0	0	1
X_1 1	1	0

$EXOR = X_1 \oplus X_2$

(This definition holds for both linear words and square (array) words.)

is often used to find out if two memory words are equal; if they are equal the result will be 0 at each bit position of the output of a logic set denoting EXOR bit-wise on each bit of the two words. Subtracting (as with ones complement computers) one of the memory words from the other also works, but on many computers this is a slower instruction. On the other hand, some machines may not have the logic to do exclusive-OR directly.

Hypothetically, if A_1 represents the digitized X-ray image (black represents white on the X-ray film) of a lung region of a person with a given progression of lung disease (e.g., pneumoconiosis progressive massive fibrosis) and A_2 represents a worsening progression of the disease, then one wishes to say that $A_1 \leq A_2$. But, one can actually test this as true since A_1 EXOR A_2 contains at least one 1 bit. Thus, if $A_1 \leq A_2$ then A_1 EXOR A_2 contains a 1 bit. Assume that control is parallel-by-bit. However, if one considers A_1 EXOR A_2' where A_2' is such that A_1 EXOR A_2' contains as many 1 bits as A_1 EXOR A_2, then one also needs to know where the 1 bits in the difference are located if one is to rank two arbitrarily presented images. The real problem is that A_1 EXOR $A_2 = (A_1$ .AND. $C(A_2))$ .OR. $(A_2$ .AND. $C(A_1))$, where $C(A) = A$, is a symmetrical relation in the sense that $A_1 \leq A_2$ whenever $A_2 \leq A_1$. One, therefore, must take A_1 .AND. $C(A_2)$, A_2 .AND. $C(A_1)$ separately. Thus, $A_1 \leq A_2$ if there is also a 1 bit in the A_2 .AND. $C(A_1)$ image; $A_2 \leq A_1$ if there is also a 1 bit in the A_1 .AND. $C(A_2)$ image. That is, $A_1 \leq A_2$ if at least a 1 bit has been added into A_1, and $A_2 \leq A_1$ if at least a 1 bit has been deleted.

The fact that two images differ in bit positions is not enough to rank them; in order to rank them they must differ in a special "order" or "direction". Thus, there is good reason for path strategies in producing ordered selection or output.

In the problem of change detection of digitized X-ray pictures, the author has thus far considered the concept of the Boolean differences of two pictures. We will explore the attractive idea of multiple-valued logic difference as a generalization of Boolean difference; our notation will be the usual [2]. Where $x \vee y = \max(x,y)$, $x \wedge y = \min(x,y)$, and where one defines

$$D_i^j(x) = D_i(x) \wedge \bar{D}_j(x) = \begin{cases} n-1, & i \leq x \leq j \\ 0, & \text{otherwise} \end{cases}$$

Allen and Givone [1] have shown that this set of operators, $\vee$, $\wedge$, D_i^j $(i,j=0,\ldots,n-1)$, forms a functionally complete algebra.

The exclusive-OR operator which was previously introduced can be generalized as follows: $MEXOR(x,y) = |x-y|$, where x,y are multiple-valued operands and $|x-y|$ is standard absolute value after integer subtraction. However, recall that one encountered a problem in ranking pictures with the symmetrical property of exclusive-OR (using the components of EXOR instead). Further information will follow regarding the usage or emulation of MEXOR,

as via the Illiac III Emulator, PAX II.

Let P be a radix n function of variables $x_{1_1}, x_{1_2},\ldots,x_{1_k},\ldots,x_{k_1},x_{k_2},\ldots, x_{k_k}$ taking on values at positions in a square array (or square word). Then, let $P^1_{x_{a_b}} = P(x_{1_1},x_{1_2},\ldots,1,\ldots,x_{k_k})$, where P^1_x denotes the function P evaluated with $x_{a_b} = 1$.

One then discovers that the multiple-valued logic difference of the radix n function P, taken with respect to x_{a_b} and evaluated at i and j, is defined as follows:

$$D^{i_j}_{x_{a_b}} P = \text{MEXOR}(P^i_{x_{a_b}}, P^j_{x_{a_b}}), \text{ a natural extension of exclusive-OR.}$$

For our purposes, the multiple-valued logic difference of P with respect to x_{a_b} evaluated at i and j may be interpreted as being a function of the amount the grey coding level of the function P changes, without regard to sign, when the cell x_{a_b} is changed from grey level i to grey level j, or vice versa.

Let us here remark that the dynamic picture of a diseased lung (with varying degrees of density at a given coordinate), when digitized, is indeed such a function P, since the x_{a_b} - values change with time (between any two respective X-rays) depending on the current state of a density as detected in grey code level changes.

Note that the multi-valued difference can either be a constant or another radix n function, as the x's change in grey levels with time.

For our picture processing application an upper bound for n is $n \leq 256$.

If the logic difference applying MEXOR is a non-zero constant Δg, $0 \leq \Delta g \leq n-1$, then the Picture function P is unconditionally dependent on the grey code value of cell x_{a_b} at grey level i and grey level j. A dynamic change in the Picture P at x_{a_b} from the one level of greying to the other will cause the rating or interpretation of the grey value of the picture function P to change by Δg grey levels (grey level units) regardless of the grey level values of the other cells $x_{1_1},\ldots,x_{a_{b+1}},\ldots,x_{k_k}$. If the logic difference is equal to 0, then P is unconditionally independent of the grey level values of the cell x_{a_b} at i and j; a grey level change in x_{a_b} from the one grey level to the other will cause no change (i.e., a change of zero grey levels) in the value of picture function P regardless of the grey values of the other cells.

Now for a very important application. The difference between "unconditional dependence" and "unconditional independence" of picture functions P is dramatically illustrated by considering variations in the rating or interpretation of light-dark tonal shading qualities of X-ray film development. Let the pair (P_1,P_2) represent the interpretation of two X-ray photos of an individual's lung taken at two different times. In one case the development of the film pair (P_1,P_2) leads to a darkened set of pictures (due to chemistry, lighting, photography, etc.) (P'_1,P'_2), and in the second case the development of the film pair (P_1,P_2) leads to a more lightened set of pictures (due to differing chemical, lighting, photographic, etc. processes) (P''_1,P''_2). An example follows (see Fig. 3.).

The following assumption is made by many radiologists (DHEW-NIOSH, CDC-99-74-4, Personal Communication):

$$\text{MEXOR}(P'^i_{1_{x_{a_b}}}, P'^j_{2_{x_{a_b}}}) = \Delta g = \text{MEXOR}(P''^j_{1_{x_{a_b}}}, P''^j_{2_{x_{a_b}}}) \neq 0,$$

and the interpretation of the difference of the two pictures P_1, P_2 is constant at x_{a_b} coordinate in the sense that variations in development do not effect the interpretation of the picture difference; thus, picture function P, such that $P(t_1) = P_1$ and $P(t_2) = P_2$, is "unconditionally dependent" on the X-ray grey level of cell x_{a_b} at grey level i and grey level j, but not dependent on variations in development, where t is a time parameter.

And, another assumption is, also, made by many radiologists (DHEW-NIOSH, CDC-99-74-4, Personal Communication):

$$\text{MEXOR}(P'^{i}_{1_{x_{a_b}}}, P'^{j}_{2_{x_{a_b}}}) = \Delta g = \text{MEXOR}(P''^{i}_{1_{x_{a_b}}}, P''^{j}_{2_{x_{a_b}}}) = 0.$$

And, the lack of any difference interpretation of the two pictures, P_1, P_2, as indicated by $\Delta g = 0$, at the x_{a_b} coordinate is taken to mean that variations in development do not effect the conclusion that the pictures are the same at x_{a_b}; thus, picture function P is "unconditionally independent" of the grey level of the cell x_{a_b} at i and j, but also not dependent on variations in development.

The remaining case is that for which the interpretation of the multiple-valued logic difference is a radix n function of the (cells) variables $x_{1_1}, x_{1_2}, \ldots, x_{k_k}$. Thus, the solution of the multiple-valued logic difference equation $D_{x_{a_b}} P = \text{constant}$, if it exists, will produce those logical conditions (logic values) necessary for a change of x_{a_b} from i to j (or symmetrically from j to i) in order to produce a corresponding change of c = constant units in the interpretation value of the picture function P.

One possible logic design for the n=3 valued multiple-valued exclusive-OR, MEXOR, is as follows with the disjoint system of algebra:

$$\text{MEXOR}(x,y) = 1 \cdot [C_0(x) \cdot C_1(y) \vee C_1(x) \cdot C_0(y) \vee C_1(x) \cdot C_2(y) \vee C_2(x) \cdot C_1(y)] \vee [C_0(x) \cdot C_2(y) \vee C_2(x) \cdot C_0(y)].$$

And, its wiring logic diagram appears in figure 4.

The functioning of MEXOR can, although quite slow, be closely approximated by the macro-subroutines ABSUBT and ADDSUB of the PAX II Emulator for Illiac III.

P'_1

0	0	0	0
0	1	2	1
0	1	2	1
0	0	0	0

P'_2

0	0	0	0
0	2	3	2
0	2	3	2
0	0	0	0

P''_1

0	0	0	0
0	0	1	0
0	0	1	0
0	0	0	0

P''_2

0	0	0	0
0	1	2	1
0	1	2	1
0	0	0	0

K = 4, n = 4

$$P(x_{1_1},\ldots,x_{1_k},\ldots,x_{k_1},\ldots,x_{k_k}) = x_{2_3},\ P^2_{x_{2_3}} = 2,\ P^3_{x_{2_3}} = 3,\ D^2_{x_{2_3}}{}^3P = 1,$$

$$\text{MEXOR}(P'^2_{1_{x_{2_3}}},\ P'^3_{x_{2_3}}) = 1 = \Delta g = \text{MEXOR}(P''^2_{1_{x_{2_3}}},\ P''^3_{2_{x_{2_3}}})$$

independent of $x_{1_1}, \ldots, x_{1_3}, \ldots, x_{2_2}, x_{2_4}, \ldots, x_{4_4}$.

FIGURE 3. Two Different Pairs of Images.

The multi-valued Picture Difference array function $\text{MEXOR}(P_1,P_2) = |P_1 - P_2|$, where P_1,P_2 are pictures can be emulated by the PAX II Emulator[1] for Illiac III[2] subroutine facility ABSUBT. The subroutine ABSUBT(IPO,NP,IPI,N1,IP2,N2) operates by $|\text{IP1} - \text{IP2}| \longrightarrow \text{IPO}$ where $N2 \leq N1$, $N1 \leq NP$. The PAX II (Illiac III) stack IP2 is subtracted from the PAX II stack IP1. The absolute value of the subtraction is placed in stack IPO, and the sign of the subtraction is placed in PAX II plane ISIGN if entry SUBSIN is, also, used.

For the case of 3-valued picture differencing, the emulation would be that of the hardware multi-valued logic suggested in figure 4.

Output Figure 1 below is the initial array configuration of a 16-valued logic array plane. The linear chain ordering is as follows: $0 \leq 1 \leq 2 \leq 3 \leq 4 \leq 5 \leq 6 \leq 7 \leq 8 \leq 9 \leq A \leq B \leq C \leq D \leq E \leq F$. Output Figure 1, also, includes the first application of MEXOR to IPI1 and IPI2. Output Figure 2 shows the final two successive left justified applications of MEXOR. An interesting geometrical property is suggested in the final output of figure 2 involving a symmetrical triangle whose rows are the symmetrical increasing and descreasing finite sequences of the even integers (or, if you prefer, logic values).

Output Figure 3 is a listing of the MEXOR code on PAX II for this particular run or emulation.

One of the fundamentals that accounts for the interesting final output pattern of figure 2 for MEXOR is the following: For any two non-negative integers a,b(or, if you will, logic values), for the absolute value function,

$$\Big|\,\big|\,|a - b| - a\,\big| - b\,\Big| = \begin{cases} 0, & \text{if } a > b \\ 2a, & \text{if } b \geq 2a \\ 2(b - a), & \text{if } b < 2a \end{cases}$$

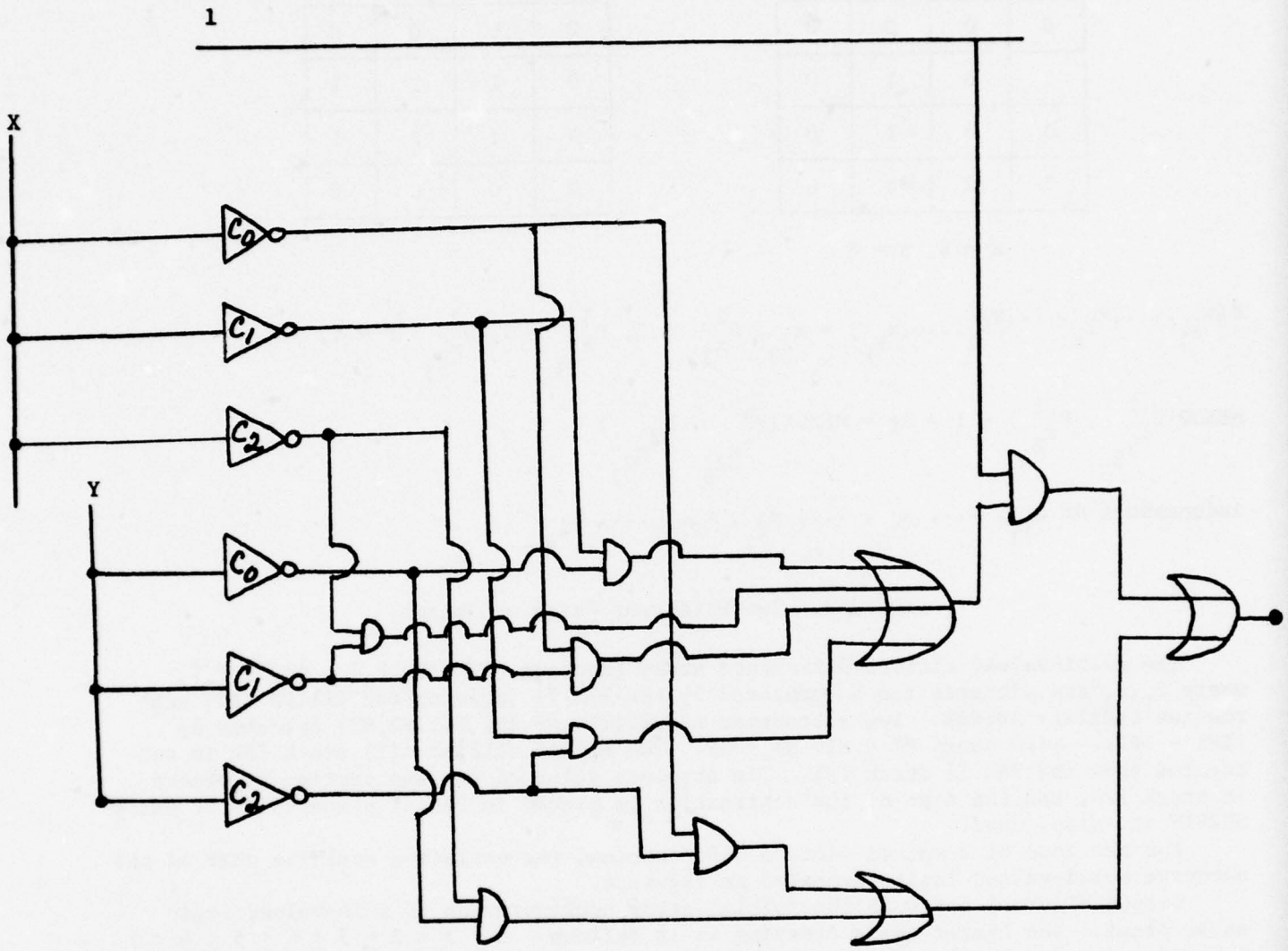


FIGURE 4. Logic Design of the 3-Valued MEXOR.

Basic elements of an associative memory computer system include an associative memory store (AM), tag stores, an instruction register (IR), a scratch pad (random-direct access for register dependent arithmetic routines) memory (SP), a comparand register (CR), control, control memory, sequential controller, and I/O. The basic instruction cycle is, if instructions are also stored in AM, (1) find the first responder (This corresponds to the parallel array instruction "FIND" for the STARAN MACRO-APPLE INSTRUCTION which finds the first responser, Y response store bits set, where APPLE=Associative Processor Programming Language) and copy its contents into the IR; (2) move the activity forward one position; (3) execute the instruction in IR.

The selection-response logic for retrieving "information" from the AM can be conceived of, initially, in the following way: we are trying to design logic for interrogators of associative memories. Stepping back one step further, for understanding, we are trying to construct a production system (PS) which is a scheme for specifying an information "processing" system, in this paper for two-dimensional information; a PS consists of a set of

productions, each production consisting of a condition (select or match criteria) and an action (response, instruction execute, data output). A PS has a collection of data structures (e.g. arrays): expressions that encode the information upon which the PS works-on which the actions operate and on which the conditions can be determined to have some of a given set of truth values (e.g., 0.1. etc.). One first considers serial production systems, and then parallel production systems.

A very important example of such an information processing system is an AM system [11, 12]. In AM systems, the memory is organized according to content, so storage and retrieval depend on relationships (equal, match, less than, greater than, similar, between limits, etc.) between data entries, i.e. WHAT data is as opposed to WHERE data is.

Also, with the possibilities of multiple-valued logic one might consider quite a variance in the selection of logic used in interrogating the AM Positive reasons for exploring this include (1) further possible reduction in program complexities, (2) more efficient potential storage and encoding of data, (3) possible reductions in wiring complexity. Negative reasons, though, include the possibility that serial response time could be slower against words of AM, unless the shorter possible word length offsets any slow serial operation times.

Traditionally, there are various ways of selecting logical rules for determining the conditions for, the design properties for, our sequential production systems - such as Markov Algorithms and conditional expressions, similar to those of LISP processing). Let us now describe conditional expressions (these arise naturally from production systems.):

(1) "$\longrightarrow$" is a syntactic symbol;
(2) $a,b,c,d,e,f,\ldots,a_1,\ldots,a_2,\ldots$ denote variable names;
(3) $a \longrightarrow b$ is a conditional expression;
(4) if C and D are conditional expressions, then $C \longrightarrow D$ is a conditional expression;
(5) if $A_1,A_2,A_3,\ldots,A_n$ are conditional expressions, then the sequence

$$\left\langle A_i \right\rangle_{i=1}^{n} = \left\langle A_1,A_2,A_3,\ldots,A_n \right\rangle \text{ is a conditional expression;}$$

(6) nothing is a conditional expression unless it follows from (1) - (5).

The proof of Theorem 1 below is straightforward from Rine [11, 12]. The theorem states that any conditional expression $\text{cond}_m(s_1,\ldots,s_m,r_1,\ldots,r_m) = \langle s_1 \longrightarrow r_1,\ldots, s_m \longrightarrow r_m \rangle$ of 2m variables, whose condition terms s_i and action (response) terms r_i are variables and which take on multiple values from the lattice $f_0,f_1,\ldots,f_{n-2}, f_{n-1}$ (a multiple-valued conditional expression), can be built up from (functions the same way as) operators of an n-valued Post algebra L [2] in several different ways depending on how the interrogation or valuation (val) condition is defined. Thus, each such conditional expression is an interrogation logic device of multiple values. And, thus, the interrogation logic, being conditional, for AM's can be so designed.

THEOREM 1. Let $f_0 \leq f_{i_1} \leq f_{j_1} \leq f_{n-1}$, $f_0 \leq f_{i_2} \leq f_{j_2} \leq f_{n-1},\ldots,f_0 \leq f_{i_\alpha} \leq f_{j_\alpha} \leq f_{n-1}$ be intervals over $f_0 \leq \ldots \leq f_{n-1}$, subintervals over the n values of the logic. Let $x \equiv \langle s_1 \longrightarrow r_1, s_2 \longrightarrow r_2,\ldots,s_m \longrightarrow r_m \rangle$ be a conditional expression of 2m variables such that val $(\bar{x})$ = ($\min_{1 \leq k \leq M} \{r_k \mid f_{i_1} \leq \text{val}(s_k) \leq f_{j_1}$ or $f_{i_2} \leq \text{val}(s_k) \leq f_{j_2}$, or ... $f_{i_\alpha} \leq \text{val}(s_k) \leq f_{j_\alpha}$). Then, one has the logical equivalence

$$x = \bigvee_{j=1}^{m} (r_j \wedge \{C_0[\bigvee_{k=1}^{j-1} ((D_{i_1}(s_k) \wedge \bar{D}_{j_1}(s_k)) \vee (D_{i_2}(s_k) \wedge \bar{D}_{j_2}(s_k)) \vee \ldots \vee$$

$$D_{i_\alpha}(s_k) \wedge \bar{D}_{j_\alpha}(s_k))]\} \wedge ((D_{i_1}(s_j) \wedge \bar{D}_{j_1}(s_j)) \vee \ldots \vee (D_{i_\alpha}(s_j) \wedge \bar{D}_{j_\alpha}(s_j))),$$

where $C_0,\ldots,C_{n-1}$ are the unary operators of the Post algebra L, $D_i(y)$ =

IPI1

```
0123456789ABCDEF
0123456789ABCDEF
0123456789ABCDEF
0123456789ABCDEF
0123456789ABCDEF
0123456789ABCDEF
0123456789ABCDEF
0123456789ABCDEF
0123456789ABCDEF
0123456789ABCDEF
0123456789ABCDEF
0123456789ABCDEF
0123456789ABCDEF
0123456789ABCDEF
0123456789ABCDEF
0123456789ABCDEF
```

IPI2

```
0000000000000000
1111111111111111
2222222222222222
3333333333333333
4444444444444444
5555555555555555
6666666666666666
7777777777777777
8888888888888888
9999999999999999
AAAAAAAAAAAAAAAA
BBBBBBBBBBBBBBBB
CCCCCCCCCCCCCCCC
DDDDDDDDDDDDDDDD
EEEEEEEEEEEEEEEE
FFFFFFFFFFFFFFFF
```

MEXOR(IPI1,IPI2)

```
0123456789ABCDEF
10123456789ABCDE
210123456789ABCD
3210123456789ABC
43210123456789AB
543210123456789A
6543210123456789
7654321012345678
8765432101234567
9876543210123456
A987654321012345
BA98765432101234
CBA9876543210123
DCBA987654321012
EDCBA98765432101
FEDCBA9876543210
```

Output FIGURE 1.

```
MEXOR(MEXOR(IPI1,IPI2),IPI1)

     0000000000000000
     1111111111111111
     2022222222222222
     3113333333333333
     4202444444444444
     5311355555555555
     6420246666666666
     7531135777777777
     8642024688888888
     9753113579999999
     A864202468AAAAAA
     B9753113579BBBBB
     CA864202468ACCCC
     DB9753113579BDDD
     ECA864202468ACEE
     FDB9753113579BDF

MEXOR(MEXOR(MEXOR(IPI1,IPI2),IPI1),IPI2)

     0000000000000000


     0000000000000000
     0200000000000000
     0220000000000000
     0242000000000000
     0244200000000000
     0246420000000000
     0246642000000000
     0246864200000000
     0246886420000000
     02468A8642000000
     02468AA864200000
     02468ACA86420000
     02468ACCA8642000
     02468ACECA864200
     02468ACEECA86420
```

Output FIGURE 2.

Output Figure 3.

```
FORTRAN IV G LEVEL   21                    MAIN                DATE = 74310      20/01/10

            C
            C       THIS PROGRAM TESTS THE MULTI-VALUED ARRAY SUBROUTINE MEXOR
            C
            C       THIS RUN IS FOR 16-VALUED LOGIC  0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
            C
0001                INTEGER IPO(5),IPI1(8),IPI2(4),IW(4),IR(16)
0002                DATA IW/1,1,16,16/, IPO,IPI1/13*0/
0003                DATA IR/0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15/
0004                EQUIVALENCE (IPI1(5),IPI2(1))
0005                COMMON /LTX/LT(32)
0006                DATA LET/1HO/
0007                LT(1)=LET
0008                CALL INPUT(8,IW,IPI1,IR,JGT)
0009                GO TO 30
0010              10 DO 20 I=1,16
0011              20 IR(I)=IR(I)+16
0012              30 CALL INROW
0013                 GO TO (10,40),JGT
0014              40 WRITE(6,50)
0015              50 FORMAT(1H1,64X,4HIPI1//)
0016                 CALL PRINT(4,IW,IPI1)
0017                 WRITE(6,60)
0018              60 FORMAT(1HO,64X,4HIPI2//)
0019                 CALL PRINT(4,IW,IPI2)
            C
0020                 CALL ABSUBT(IPO,5,IPI1,4,IPI2,4)
            C        MEXOR=ABSUBT
0021                 WRITE(6,70)
0022              70 FORMAT(1HO,55X,16HMEXOR(IPI1,IPI2)//)
0023                 CALL PRINT(5,IW,IPO)
0024                 CALL ABSUBT(IPO,5,IPO,5,IPI1,4)
            C        MEXOR=ABSUBT
0025                 WRITE(6,80)
0026              80 FORMAT(1HO,52X,28HMEXOR(MEXOR(IPI1,IPI2),IPI1)//)
0027                 CALL PRINT(5,IW,IPO)
0028                 CALL ABSUBT(IPO,5,IPO,5,IPI2,4)
            C        MEXOR=ABSUBT
0029                 WRITE(6,90)
0030              90 FORMAT(1HO,48X,40HMEXOR(MEXOR(MEXOR(IPI1,IPI2),IPI1),IPI2)//)
0031                 CALL PRINT(5,IW,IPO)
0032                 STOP
0033                 END
```

$\bigvee_{t=i}^{n-1} C_t(y)$, and $\overline{D}_i(y) = \bigvee_{t=0}^{i} C_t(y)$.

Let us consider the AM system components consisting of an AM, a CR, a mask register (MR), and a response store (RS). Suppose that the MR is masking out all of CR, for the AM search, except for field F_a. Let us consider search type that is serial by word and serial by bit. A single search cycle, for a given cell (bit) of F_a, examines every corresponding cell position for every word (square words for picture processing) of AM and returns a value of n-1 (using n-valued logic) to the corresponding RS cell if the search is successful or returns a value of 0 to the corresponding RS cell if the search fails. In this type of search all successful responses are retrieved in at most length (F_a) cycles. It would take one cycle if the search type were parallel by word and parallel by bit (STARAN functions serially by bit).

Let us take a more specific look at the search process. Let n=9 and search the AM store for all those 3-cell numbers that are equal to or less than 550; we place 550 in the CR. The use of $\overline{D}_i$ gates for the CR design should make the search faster than using the C_i gates. By using the $\overline{D}_i$ gates and a serial by cell type search, at most 3 passes or cycles are necessary for retrieval, the final state of RS being as indicated below.

Let us take a more specific look at the search process for "betweenness".

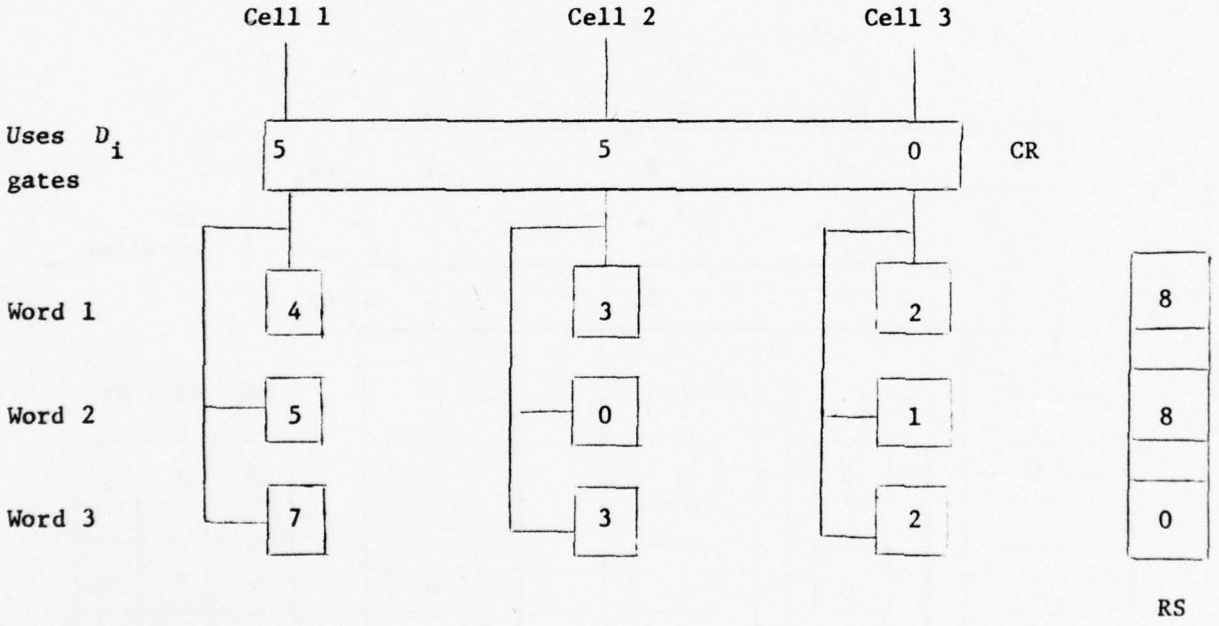


FIGURE 5.

Referring to fig. 6, let n=9 and search the AM store for all those 4-cell numbers that are equal to or greater than 3 3 4 1 and equal to or less than 5 5 5 0; one places 3 3 4 1 in the CR, and 5 5 5 0 in the CR_2; CR_1 and CR_2 are comparand registers that function in parallel. The use of D_i gates for the CR_1 design and $\overline{D}_i$ gates for the CR_2 design will make the actual search faster than, that is fewer cycles, using the C_i gates. By using D_i, $\overline{D}_i$ gates and a serial by word serial by cell type search with CR_1 and CR_2 functioning in parallel, at most 4 passes or cycles are necessary for retrieval the final response state due to the CR_1 search is in RS_1, the final response state due to the CR_2 search is in RS_2, and the AND of RS_1 and RS_2 is in RS.

An alternate design would be to eliminate RS, and put $RS_1 \wedge RS_2$ in either RS_1 or RS_2.

Another alternate design would involve the replacing of the gating functions D_i and $\bar{D}_i$ in the two comparand registers CR_1 and CR_2 by a single gate function D_i^j such that $D_i^j = D_i \wedge D_j$ (functionally equivalent) for a single comparand register CR.

By observing the result of theorem 1, one makes the following observation: Let there be c cells for each word in AM; let there be distinct intervals to be tested for the AM store; if search is serial by word and serial by cell and response time is to be dependent on c, and if $D,\bar{D}$, gates are to be used separately in designing the comparand registers, then it will sometimes be necessary to use as many as 2^α comparand registers $CR_1, \ldots, CR_{2^\alpha}$. If one is willing to enlarge the number of cycles necessary for search, then one can reduce the number of comparand registers back to 2. The third alternative would be to construct a very complicated comparand register displaying simultaneously the functional aspects of all $D_{i_1}, \bar{D}_{j_1}, \ldots, D_{i_\alpha}, \bar{D}_{j_\alpha}$. Hence, there is a tradeoff between the number of distinct comparand registers, and the functional complexity of a given comparand register (obtained from the complexity of the application relation).

By changing the result of theorem 1 to the corresponding result of a parallel by word search type, then the number of search cycles necessary for response (selection) can be divided by a factor of c, assuming parallel by cell.

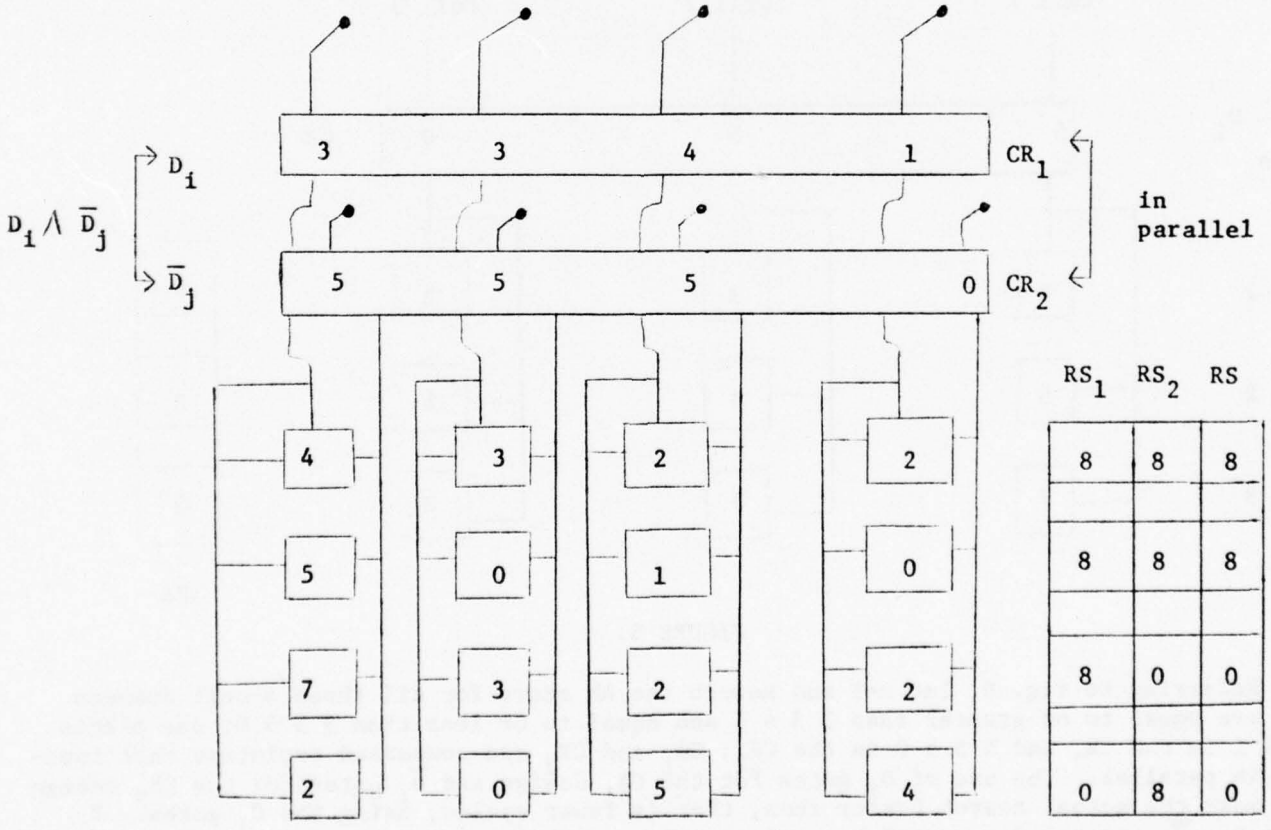


FIGURE 6.

Also, the search logic itself for parallel search would be simplified from that of theorem 1 to just

$$\bigvee_{j=1}^{m} (r_j \wedge ((D_{i_1}(s_j) \wedge \bar{D}_{j_1}(s_j)) \vee \ldots \quad \ldots \vee (D_{i_\alpha}(s_j) \wedge \bar{D}_{j_\alpha}(s_j))).$$

Let us now state a second theorem having the same hypothesis as theorem 1 and whose proof is a straightforward generalization of the proof technique employed in Rine [11,12].

THEOREM 2. Assume the hypothesis of theorem 1. Let "X" denote a logic operator which can, for a given position, be either $\wedge$ or $\vee$. Then one has the logical equivalence

$$x = \bigvee_{j=1}^{m} (r_j \wedge C_0 \bigvee_{k=1}^{j-1} (D_{i_1}(s_k) \wedge \bar{D}_{j_1}(s_k)) \,(X)_1\, (D_{i_2}(s_k) \wedge \bar{D}_{j_2}(s_k)) \,(X)_2 \cdots (X)_{\alpha-1}$$

$$(D_{i_\alpha}(s_k) \wedge \bar{D}_{j_\alpha}(s_k)) \wedge ((D_{i_1}(s_j) \wedge \bar{D}_{j_1}(s_j)) \,(X)_1 \cdots (X)_{\alpha-1} (D_{i_\alpha}(s_j) \wedge \bar{D}_{j_\alpha}(s_j))),$$

where each $(X)_a$ is either $\wedge$ operator or $\vee$ operator.

NOTE: There should be a "subscript" on "little x" corresponding to the various assignments at the (X).

References

1. Allen, C. M. and Givone, D. D., "Final report on design of multiple-valued logic systems", The Digital Systems Lab of the Electrical Engineering Department, State University of New York at Buffalo, Buffalo, N. Y., June, 1970.
2. Epstein, G., "The lattice theory of Post Algebras", Trans. A.M.S. 95 (1960), 300-317.
3. Goodyear Aerospace Corporation, STARAN-S project reports, GAC, Akron, Ohio, 1972-74.
4. Hanlon, A. G., "Content - addressable and associative systems - a survey", IEEE Trans. Computers, (August 1966), 509-521.
5. Harrison, M., "Introduction to Switching and Automata Theory", McGraw-Hill Series in Systems Science, New York, 1965.
6. ILLIAC III, Illiac III technical project reports, Computer Science Dept. and Coordinated Sciences Lab., The University of Illinois, Urbana, Illinois.
7. Illiac III Reference Manual, Volumes I and II, edited by B. H. McCormick and B. J. Nordmann, Dept. of Computer Science, University of Illinois, 1971.
8. King, W. K., "Design of an associative memory", IEEE Trans. Computers, (June 1971), 671-674.
9. Lillestrand, R. L. and Hoyt, R. R., "The design of advanced digital image processing systems", Photogrammetric Engineering (1974), 1201 - 1218.
10. Radosevic, R. G., Bruno, C., Adams, N.J., and Jun, J., "Associative array processing for topographic data reduction", Goodyear Aerospace Corporation, Akron, Ohio, Interim Technical Report (STARAN), Feb., 1974.
11. Rine, E. C., "Conditional and Post algebra expressions", Discrete Mathematics Vol. 10 1974, 309 - 323.
12. Rine, D. C., "Conditional and Post algebra expressions", Proceedings of the 1973 International Symposium On Multiple-Valued Logic, Toronto, May 1973, 173 - 191. Also, "A note on interrogation logic of associative memories", submitted to IEEETC, 1974.
13. Seeber, R. R. and Lindquist, A. B., "Associative memory with ordered retrieval", IBM Journal of Research and Development 6 (1962), 126-136.

APPLICATIONS OF FUZZY LOGIC TO MEDICAL DIAGNOSIS

Harry Wechsler
Information and Computer Science Department
University of California, Irvine

1. INTRODUCTION

Automated medical diagnosis has been of continual interest to a variety of people. A lot of work has been done in the field, but except for the work of Shortliffe et al [8], much of it is technically based on statistical decisions models, and the results were not too encouraging for further investigation.

A summary of diagnostic models is presented by Craft [14], and by far the Bayes Criterion is the most common. It is based on the notion that the best set of pattern assignments is the one resulting in the least expected probability of misclassification. This criterion has been used by Garry and Barrett [16]. Craft concludes his paper showing that the accuracy obtained by using several different statistics was little better than 50-60%. He concludes that further refinements of the statistical approach were unlikely to provide a major improvement and that the diagnosis models should point a new direction.

The design of all the systems presented by Craft share the following weaknesses:

a) the statistical dependencies require acquisition of high order conditional probabilities;

b) it is hard to get a good data base, i.e., to get the probability of any disease with respect to any subset of symptoms;

c) there is no way of adding procedural knowledge.

The knowledge such a system can have can be expressed either in a static form, a descriptive one or in a constructive form, a procedural one. A descriptive knowledge base system will merely look for the probability of having a specific disease given a set of symptoms and will choose the diagnosis based on a likelihood approach.

In contrast, a procedural knowledge base is composed of a set of programs which are organized to test for a cluster of symptoms, a syndrome or a disease. Medical knowledge of what symptoms should be considered after a particular set are found is organized as a body of executable code -- a procedure for systematically reasoning about the sets of symptoms attributed to the patient being diagnosed. In this paper we advocate a variety of this approach.

The proposed approach is going to have the following advantages:

a) diagnostic information is represented procedurally;

b) using procedures to explicitly deal with statistically dependent symptoms through use of boolean combinations, as

(AND (S_1 (OR S_2 S_3))) where $S_{i = 1, 2, 3}$ stands for symptom i;

c) adding new information via change (or extension) of procedures rather than through building a large data base to improve the statistical decision rules;

d) allowing Inexact Concepts (Multi-Valued) to deal with degrees of a symptom;

e) allowing the interpretation of inexactness to vary with context.

These last two advantages are the unique features of the diagnostic system proposed here.

At this point it is appropriate to state how this approach is related to the one developed by Shortliffe et al, given that his system, the MYCIN system, is the closest one to our proposed system. In MYCIN, the procedural knowledge is represented as production rules (P,A) where the action A is taken if the premises P are fulfilled. No provisions are made for a full goal-oriented approach (this is a variety of heuristic search) [3], where one decision-program for a specific disease can contain as one of its steps (statements) a call for another decision-program. Shortliffe's paper states the following:

> "Just as Bayesians who use their theory wisely must insist that events be chosen so that they are independent (unless the requisite conditional probabilities are known), we must insist that dependent pieces of evidence be grouped into single rather than multiple rules. The system therefore becomes unworkable for applications in which large numbers of observations must be grouped in the premise of a single rule in order to insure independence of the decision criterion. In addition we must recognize logical subsumptions, when examining or acquiring rules and thus avoid counting evidence more than once (For ex: if $S_1 \rightarrow S_2$, then the confirmation of the hypothesis H_1 with respect to the symptoms S_1 & S_2 is equal to the confirmation of H_1 with respect to S_1, regardless of the value/confirmation of H_1 with respect to S_2). MYCIN does not know this."

Therefore, MYCIN system represents only a first step toward a general goal-oriented approach. The way new information is added in MYCIN is the one to be also used in our approach. The inexact reasoning used by MYCIN is a measure of belief/disbelief rather than the use of fuzzy concepts. Each production rule (P,A) has an associated certainty factor that reflects the measure of belief/disbelief of the expert who suggested the rule. In our approach we allow various predicates e.g. "blood pressure" to take on a range of values. Thus, symptoms sets can be expressed in a fuzzy logic. In the following paragraphs we are going to present the use of fuzzy concepts, first informally, and eventually to axiomatize the mechanics of fuzzy computation. Then we show how a fuzzy diagnostic system should look like.

2. MECHANICS OF FUZZY COMPUTATION

Computers perform their tasks following the way they are instructed and they will continue to do so. The purpose of this paragraph is to show how we expect to get better performance from computers and a behavior more similar to that exhibited by people. This behavior is going to be a fuzzy behavior and by it we mean that the instructions to be executed deal with 'fuzzy concepts.' This approach, hopefully, will diminish the gap between computers and people and what is more important it will improve their performance, leading eventually to build systems that can reason effectively with 'inexact' concepts. We hope that the way we can get all the above at the present time is by incorporating in one of the new programming languages, e.g., QA4[3], fuzzy concepts of behavior. The first step in this direction has been taken by Kling [7].

The prevailing attitude in contemporary scientific research is to get exactness as much as possible. But, as Zadeh writes in his report [6],"a wise step can be today, less preoccupation with exact quantitative analyses and more acceptance of the pervasiveness of imprecision in much of human thinking and perception." He believes that by accepting this reality rather than assuming that the opposite is the case, we are likely to make more real progress in the understanding of the behavior of humanistic systems than it is possible within the confines of traditional methods. The effectiveness of a problem-solving system depends on the ability to provide it with a symbolic representation of the environment in which it is called to operate, and with heuristics which control the action of the problem solving system depending on such a knowledge representation [2].

One of the most exciting problem solving system is the Question Answering System (QA) [2]. Such a system must store a lot of data which can be relevant in different ways to different queries. Keeping in mind the finiteness of our resources and being willing to get efficiency as much as possible, we would like to express the way the data is relevant with respect to different criteria, as concisely as possible. Before we present our ideas about designing good QA systems according to these concepts, we think it worthwhile to elaborate our description of imprecise concepts.

The concept of information is based on the concept of probability. The concept of probability is defined on a σ-algebra or a distributive lattice. All the laws of Boolean logic can be derived from the characteristics function f(A/a) which is 1 if object 'a' belongs to class A and it is 0 if 'a' does not belong to class A. A class is understood as the extension of a predicate. The basic postulate that 'a' and A determine the value of f(A/a) which is either 0 or 1 is called "the postulate of fixed truth set." The breakdown of this postulate was already noticed when philosophers made distinction between the primary quality and secondary quality. The characteristic function of a secondary quality depends not only on 'a' and A but also on a third argument x, which is the observer [4]. At the same time it must be clear to the reader, that in the majority of cases, we cannot reduce the continuous value of f(A/a) to a binary case.

A fuzzy subset A of a universe of discourse U is characterized by a membership function $m_A : U \rightarrow [0, 1]$ which associates with each element 'u' of U a number $m_A(u)$ in the interval [0,1], with $m_A(u)$ representing the grade of membership of u in A [1]. Kling in his report [7], uses these concepts relating them to PLANNER, a procedural language developed at MIT [20].

Now, let us ask some very simple questions. Are we really interested in the grade of membership of each of A's members? What will happen if some day the definition of f(A/a) is going to be changed? Are we going to check in our storage all previous members of A, change their membership grade, and maybe exclude some of them? And this is not all that must be done. What about possible new members of A,(i.e., data for which f(A/a) > 0) and what about different membership grade with respect to different observers x? Are we going to search all the data? From all these difficulties we believe that using the membership grade is impractical for an efficient QA system.

In the following we will try to take advantage of the facilities provided by QA4 [3][1] and to show how it is possible to build a QA system which will incorporate in itself the imprecision-fuzziness concepts discussed above. Rather than using membership grade we are going to define classes of generative functions, whose task will be to define implicitly fuzzy sets.

A class is related to a concept and it contains one or more functions f_i^P (P stands for the property to be checked), each of them related to some context (i.e., to the observer x_i). The context facility is provided by QA4, so no special problems will arise from the above definitions. Now, if we change our mind about one concept, all that must be done is to change the definition for some function f_i^P. If we think about properties we can distinguish two categories:

a) The first category contains those properties which can be measured (example: height, weight, age). These will be called primitive properties.

b) The second category contains those properties which can not be measured and can be usually expressed only as functions of primitive properties (example: ripeness, freshness)[2]. The extension of this definition to a recursive one is obvious, i.e., if

Property b_1 = f (properties $\vec{a}$)

Property b_2 = f($\vec{a}$, b_1) is well defined.

---

[1] The QA4 language has features that are recognized as useful for problem-solving programs; these features include built-in backtracking, parallel processing, pattern matching, and set manipulation. Expressions are put into a canonical form and stored uniquely, so that they can have property lists. A context mechanism is provided, so that the same expression can be given different properties in different contexts. The QA4 interpreter is implemented in LISP and can interface with LISP programs. QA4 can store information in an imperative form, as a program. This makes it possible to store advice locally rather than globally; in giving information to the system we can tell it how that information is to be used [3].

[2] This classification is not at all arbitrary. As Boole wrote: "All logical propositions may be considered as belonging to one or the other of two great classes, to which the respective names of 'PRIMARY' or 'CONCRETE PROPOSITIONS' and 'SECONDARY' or 'ABSTRACT PROPOSITIONS,' may be given. In other words, every assertion either expresses a relation among things and therefore it is CATEGORICAL or it expresses a relation among propositions and therefore it is HYPOTHETICAL in its nature.

How are we going to deal with properties P which belong to the category (a)? In other words, how $F^P = \{f_i^P\}$ are defined?

$$f_i \triangleq f_i : H^P \times A^P \rightarrow \tilde{u} \in U$$

where H^P and A^P stand for the hedge[3] and adjective domain respectively and $\tilde{u}$ is some subset of the universe of discourse. The result of applying f_i^P is to get in fact $f_i^P(\tilde{u}/a)$, i.e., the predicate $\tilde{u}$ associated with class u. By using this kind of definition we allow, in fact, that the classes defined by $\{f(\tilde{u}/a)\}$ be overlapping, and this is consistent with the imprecision concept.

- Example: P = age
  H = {null[4], very, little, ...}
  A = {old, young}
  $F^P = \{f_i^P\}$

For some i : f_i^P (null young) = f_i^P (young) = (age 20, age 30).

If A = {baby, child, teenage, young, mature, old} for some other i:

f_i^P (baby) = (age 0, age 3)

f_i^P (child) = (age 2, age 14)

f_i^P (teenage) = (age 12, age 18)

f_i^P (young) = (age 17, age 30)

f_i^P (mature) = (age 26, age 60)

f_i^P (old) = (age 55, $\longrightarrow$ [5])

The overlapping of the above sets express better the concept of fuzziness than the membership function. We delay to the sequel the way we can cope with the properties belonging to the category (b).

Another important question is: being given with two items, which of them fits better the predicate $\{f(\tilde{u}/a)\}$ assuming that both of them belong to the class.

---

[3]By HEDGES we mean words of the following type: very, more or less, quite, fairly and so on.

[4]The hedge "null" means that no hedge at all is going to be used!

[5]Any positive integer greater than 55.

(In Zadeh's terminology, given a set A and a_1, $a_2 \in A$ decide either $m_A(a_1) \geq m_A(a_2)$ or $m_A(a_2) \geq m_A(a_1)$). We think that for the above question, the following two kinds of order-decision are suitable and sufficient. The first one tells us the closer you are to the middle of the interval defined by $f(\tilde{u}/a)$ the better you fit the property.

Example: f_1^{AGE} (young) = (age 17, age 30)

Now, if we look for someone young, we will prefer someone whose age is 25 to someone whose age is 20. Let us call this order the middle order and associate with f, M such that the definition of the class and its order are represented together by $< f(\tilde{u}/a); M >$.

The second criterion tells us simply that the first member of the interval is the best, i.e., we will get the interval ordered from left to right.

Example F_1^{GRADE} (good) = (A B). Let us call this order the LB order and then, as above, we will get $< f(\tilde{u}/a); LB >$, where LB has the property that the more left you are in the interval, the better off you are.

How can we conclude the discussion of the properties belonging to the category (a)?

First of all, let us remark that the definition is built based only upon the tasks we can be posed with. There is no real meaning to the membership grade. We can be asked (related to a QA system) only two kinds of questions related to some primitive property:

(i) Does someone have the property (i.e., is someone young)?

(ii) If both a_1 and a_2 have the property, which of them fits it better?

As we have seen before, by using our definition, we are able to answer both questions. This is not the only advantage of using our definitions. By using this approach all the operations involving predicates can still be done using the set-theoretical operations, which is of course, much more naturally than deriving new definitions, as it is done in [1].

What about implementation? As in QA4 the ASSERT and DENY statements take care about the existence of an item, these constructs are the ones to be used for the existence of a particular item at a given time. (For the existence, we assume that the postulate of fixed truth set is valid.) What about properties? This is done in QA4 by using PUT statements of the following format:

(PUT syntactic-form indicator property)

Now, the PUT statement will look like the previous or the following:

(PUT syntactic-form indicator value-field function)

where we understand the value field as being non-empty, i.e. containing either a real value or an adjective preceded perhaps by a hedge; and by function we mean the function stored in the system which takes care of the mapping (and also implicity

takes care of the context). The EXISTS statement is going to be changed in the same way. The PATTERN MATCHER must be built following the same concepts. Let us develop the idea and try to match the value field set up by the PUT statement with the value field the EXISTS statement is looking for.

(i) Both value fields are real-valued. Therefore, we can use the previous mechanism of the PATTERN MATCHER for comparing them.

(ii) Both value fields are expressed by [hedge] adjective. Use the set inclusion concept. Example:

(PUT Dick age very old f_1) (EXISTS (←X man) age old)

where f_1 is some context used for the age-property and ← means the variable X is to receive a value. If we assume that before the PUT statement we have executed

(ASSERT (Dick man))

we will succeed in executing the EXISTS statement because hedge (a) $\subseteq$ a, i.e., very old $\subseteq$ old. At the same time we should report a failure for the attempt to fit old $\subset$ very old.

(iii) Suppose now, that the value field of PUT is real-valued and that of EXISTS is [hedge] adjective. Use f ([hedge] adjective) for generating the class and check if the real-value belongs to it or not.

(iv) If the value field of PUT is [hedge] adjective and that of EXISTS real-valued, we will report a failure.

Now that we have described the category (a) of properties and the way we deal with them, we will exhibit the way we can cope with the category (b) of properties, i.e., non-primitive properties. For the sake of simplicity, we are going to employ a simple example not drawn from the field of medical diagnosis. Suppose we want to find in our refrigerator a ripe fruit. We need a goal program of type CHOOSE and we know that ripeness is a function of freshness, color, and size.

Size and color are properties of category (a) type. Freshness is a property of category (b) type and it is a function of the numbers of days the fruit was kept in the refrigerator (this is a property of category (a) type).

So, f (very fresh) → (0 days, 2 days)
f (fresh) → (0 days, 4 days)
f (unlikely fresh) → (3 days, 7 days)
f (not fresh) → (6 days, → [6])

Also, as we said previously

RIPENESS = g (FRESHNESS ∩ COLOR ∩ SIZE)

What will the program look like? First of all, at some stage we have the statements:

(ASSERT ((SET FRUIT1 FRUIT2. . .FRUIT100) FRUIT))

and for each fruit j

[6] Any positive integer greater than 6.

```
(ASSERT (FRUITJ FRUIT))
(PUT (FRUITJ FRUIT) KIND APPLE)
```

The kind can be apple as above or anything else. Later during the execution:

```
(GOAL $CHOOSE (←M RIPE))
```

For performing this goal of class CHOOSE the Pattern Matcher will evluate the following λ-expression:

```
(LAMBDA (←N RIPE)
    (PROGRAM (DECLARE L1 L2)
    (EXISTS (SET ←L1 ←←L2) FRUITS)
    (EXISTS ($L1  FRUIT) SIZE NOT TOO BIG f1size)
    (EXISTS ($L1  FRUIT) FRESHNESS FRESH f1fresh)
    (EXISTS ($L1  FRUIT) COLOR RED f1color)
    (DENY ((SET FRUIT1. . .FRUIT1ØØ) FRUITS))
    (ASSERT (SET $$L2 FRUITS))
    (RETURN (TUPLE ($L1 = (GET ($L1 KIND)))))
    (DENY (($L1 FRUIT)))))
```

In the above example we used the DENY and ASSERT statements in order to modify the state of the world after we choose a ripe fruit because we look for a ripe fruit in order to eat it, and therefore it doesn't exist any more after it is found. Usually λ-expressions have variables with prefix ← in the bound variable part; if we want to refer to the bindings of those variables in the body of the lambda expressions, we use the same variables with the prefix $. The rules for matching a variable prefixed by ←← or $$ are analogous to those for ← and $ prefixes. However, variables with these prefixes will be bound to a fragment of a set [22].

3. MEDICAL DIAGNOSIS USING PROCEDURAL AND FUZZY CONCEPTS

In many ways the consulting part of a medical-diagnosis system is like making available a computer-stored textbook of medicine [13]. Let us reproduce from [12] the symptoms of ulcers:

> "In the classic syndrome the pain is felt at or slightly below the tip of the xiphoid process, spreading more to the right of the midline than to the left, and rarely rising above the xiphisternal junction. It occurs 1 to 3 hours after eating, sometimes awakens the patient at night, and is least commonly present in the morning before breakfast. It is relieved by milk and other protein foods, even though cold, by antacids,... . In gastric ulcer, pain often occurs less than an hour after eating, is less reliably relieved by food, is accompanied by nausea, and only very rarely occurs at night."

As it is easy to see from the above description, rather than dealing with probabilities of different kinds of ulcers with respect to the syndromes, we look for the presence or the absence of syndromes, which are almost all expressed in fuzzy concepts.

All of the following examples are expressed in the terminology provided by QA4 [3] and the meaning of the symbols used was described in the previous paragraph.

Looking for a possible ulcer is equivalent to the following statement in a goal-oriented environment

(GOAL $FIND (←X ULCERS))

A program of type FIND is going to search for possible candidates belonging to the ulcers set and check if the syndromes are either present or not. The ulcers' decision-making knowledge is represented by context-dependent functions as $f^{property}_{ulcers}$, whereas the gastric ulcer which is a subset of ulcers can be referred by either f^{p}_{u} or $f^{property}_{gastric\ ulcer} = f^{p}_{gu}$. A typical program is a sequence of statements which includes, among the others, jump, conditional statements and call to any other program. Based upon the description provided by Mellinkoff, we know that the location of the pain should be almost the same for different kinds of ulcers, but if a gastric ulcer is present, it should be accompanied by nausea and only very rarely occurs at night.

Therefore, the procedural-knowledge should first of all check for common characteristics belonging to all kinds of ulcers and only finally to check for a specific kind of ulcer by using a multiple-conditional statement. Under the general context of the patient being checked for ulcers, a possible program should look like:

(LAMBDA (←D ULCERS)
(PROGRAM ( )
(EXISTS Patient pain-location slightly below the top of the xiphoid process f^{p1}_{u})
(EXISTS Patient time of pain breakfast f^{p2}_{u})))

If all the above tests are positive, we know that an ulcer has been detected and we shall want to find out which one.

(EXISTS (SET ←L1 ←←L2) ULCERS)

The first choice may be gastric ulcers and if the answer to the following tests are true, we will conclude that a gastric ulcer is present.

(COND((AND (EXISTS Patient nausea $f^{nausea}_{L_1}$)
(EXISTS Patient time of pain $f^{time}_{L_1}$))
(RETURN (TUPLE (=Patient = (GET ($L1 KIND))))))

If one of the precedent EXISTS statement fails, i.e., it returns NIL, the AND is false and a next set of conditions is checked for finding what kind of ulcer we did detect.

From this brief example it is clear that there is no way to avoid the use of fuzzy concepts. In fact, the description reproduced contains almost only fuzzy concepts!

4. A PROPOSED MEDICAL DIAGNOSIS SYSTEM

Medical diagnosis is equivalent to pattern recognition and an excellent review of the field is provided by Patrick's paper [13].

Our intention is to present the difficulties encountered by previous research and to show how they can be solved using the fuzzy concepts discussed elsewhere in this paper. Patrick reviews some of the problems such systems failed to cope with. These include:

a) the measurements $x_1, x_2 \cdot \cdot \cdot x_L$ are considered to be statistically independent features when in fact they are measurements which are statistically dependent;

b) there are failures to recognize significant measurement (features) for a particular class (failure to incorporate a priori dimensionality reduction);

c) inability to introduce a priori knowledge about correlation among measurements or features.

Lusted in [15] had noted that even those researchers using Bayesian statistical inference models as an approach to medical decision-making have begun to group or cluster symptoms as a way to improve the efficiency of their mathematical search strategy.

Wortman in [17] feels that "hierarchical structures" are needed, because they will allow exponential increases in alternatives with only linear increases in time and that this will fit with Mandler's theory of memory organization [19].

Ledley [18] suggests introducing a priori medical knowledge as a class-feature relationship defined in terms of decision tables. He realizes that the decision table approach may not be appropriate when class conditional probability densities "overlap" and suggests using the Bayes theorem with multivariate class conditional probability densities, not specifying how to estimate them. As a feasible alternative he suggests storing in memory many sets of measurement values and recognizing which ones are closest to a patient's measurement vector (nearest neighbor rule). The system we are going to propose is aware of the way knowledge can be stored, retrieved and inferred, and the implications with regard to efficient use of the memory and fast-decision making. Our model is supposed to be more reliable in that it is closer to the way in which real-world medical diagnosis is performed and therefore it is more likely that the physicians will accept it, by cooperating with and being willing to improve it.

Taking in account the deficiencies remarked earlier we decided to discard the Bayes criterion as our tool. We do not choose anymore a possible disease because it is the most likely to be with respect to some cost function, but rather we are looking to see if a given disease is either present or not.

The knowledge of our system should be built around two different memory-structures corresponding to the first and second category attributes, respectively. The first storage unit is intended to contain data relevant to first category attributes and their valued-intervals within different contexts. Example: blood pressure and

fatness are such attributes. Within some defined context as heart diseases, we can define high blood pressure as being the valued-interval (17, → [7]) while for an intern physician high blood-pressure is (14, → [8]).

The decision-making knowledge is going to be stored in the second storage unit in a procedural-form (goal-oriented programs, i.e., disease-oriented programs).

For each patient we are going to keep a data-base unique to him (the ongoing record) which is used by the decision-making unit.

A top-level diagram of the proposed system should look like the one shown in Figure 1.

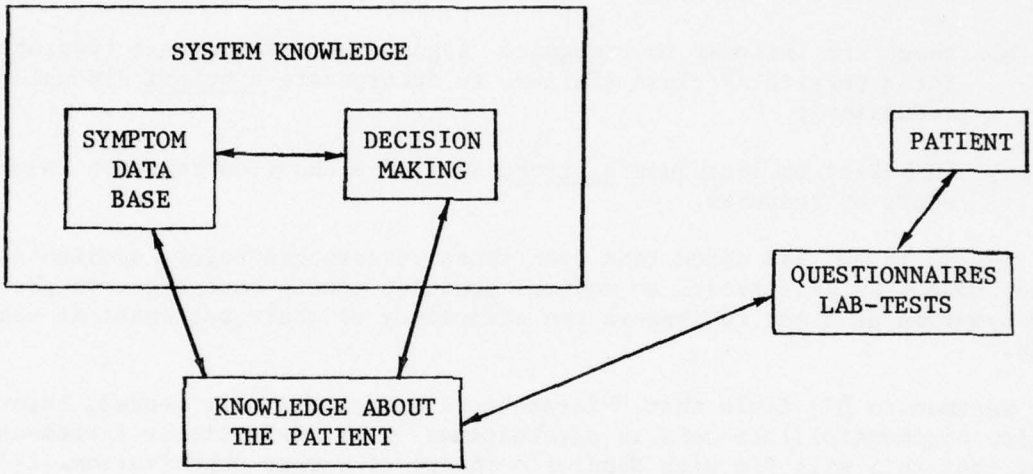


Figure 1.

The Symptom Data Base is built around the fuzzy concepts and all pertinent data of use for the decision-making is present here. More than one context is allowed to exist at a given time and the medical-diagnosis is done with respect to a given context. Contexts are created by the "decision-making" programs during the search for a solution.

The decision-making unit is organized around clusters of programs (i.e., specialists) which should be activated by demons[9]. Backtracking, a facility existing in QA4, is also used because the demons can point to different clusters of programs according to how their preconditions are fulfilled. Based upon the

---

[7]Any positive integer higher than 17.

[8]Any positive integer higher than 14.

[9]A demon is the software equivalent for hardware triggers that are fired out when some preconditions are fulfilled and force specific paths through the decision-making tree to be taken.

procedural form the knowledge is stored, sharing of procedures is made available and therefore the size of this memory unit can be kept reduced. Two of Patrick's problems were the inability to recognize significant features for a particular class (cluster) and the inability to introduce a priori knowledge about correlation among features. The first of the above problems can be solved by using a natural sequence-order of statements within the programs which are supposed to recognize some diseases. The ordering should force us to fail as soon as possible if we follow a misleading path and to backtrack to the previous point of choice.

The correlation among features can be introduced by using the IF-THAN-ELSE conditional construct. The data-base containing knowledge about the patient is a dynamic unit rather than a static one. It should contain positive evidence as well as negative with respect to the presence or the absence of symptoms. This unit should simulate a short-term memory and be used previously to any attempt to retrieve or infer something from the system knowledge.

5. CONCLUDING REMARKS

A proposed medical diagnostic system has been proposed in which the fuzzy concepts play a major role. It improves upon some of the most obvious weaknesses of the statistical decision-theoretic approaches most commonly used in automated diagnostic systems. Some of these defects can be alleviated by the use of procedural representations of the organization of symptoms into disease related syndromes. Furthermore, the use of fuzzy concepts allows additional and substantive flexibility and expressive power.

6. ACKNOWLEDGEMENT

Special thanks are given to Rob Kling for his suggestions and encouragement in writing this paper.

REFERENCES

[1] L. Zadeh, "The concept of a linguistic variable and its application to approximate reasoning," Memorandum No. ERL-M411, October 1973, Berkeley.

[2] F. Sirovitch, "Some ideas on semantic memory in automatic learning of heuristics," Carnegie Mellon University, Technical Report, 1972.

[3] J. Derkensen, J. F. Rulifson and R. A. Waldinger, "The QA4 language applied to robot planning," Proc. Fall Joint Computer Conference 1972, New York. Spartan Books, 1972.

[4] S. Watanabe, "Modified Concepts of logic, probability and information based on generalized continuous characteristic function," Information and Control 15, 1-21 (1969).

[5] A. Turing, "Computing machinery and intelligence," from "Computer and Thought," McGraw Hill, 1963.

[6] L. Zadeh, "Outline of a new approach to the analysis of complex systems and decision procsses," IEEE Transactions on Systems, Man and Cybernetics, Vol. SMC-3, No. 1, January 1973, pp. 28-44.

[7] R. Kling, "Fuzzy Planner: Reasoning with inexact concepts in a procedural problem solving language," Journal of Cybernetics (in press).

[8] E. Shortfiffe et al, "An artificial intelligence program to advise physicians regarding antimicrobial therapy," Computer and Biomedical Reserach 6, 544-560 (1973).

[9] N. J. Nilsson, "Learning machines," McGraw Hill 1965.

[10] D. Bobrow, B. Raphael, "New programming languages for A.I. research," Computer Surveys, Vol. 6, No. 3, September 1974.

[11] G. Boole, "An investigation of the laws of thought," Dover 1958.

[12] S. Mellinkoff, "The differential diagnosis of abdominal pain," McGraw Hill, 1959.

[13] E. Patrick et al, "Review of pattern recognition in medical diagnosis and consulting relative to a new-system model," IEEE Trans. on Systems, Man and Cybernetics, Vol. SMC-4, No. 1, January 1974.

[14] D. Croft, "Is computerized diagnosis possible?" Computer and Biomedical Research 5, 351-367 (1972).

[15] L. Lusted, "Introduction to medical decision making," Charles C. Thomas, Springfield, Ill. 1968.

[16] G. Gorry, G. Barnett, "Experience with a model of sequential diagnosis," Computer Biomedical Research, Vol. 1, pp 490-507, May 1968.

[17] P. Wortman, "Medical diagnosis: An Information Processing Approach," Computer Biomedical Research, Vol. 5. pp. 315-328, August 1972.

[18] R. Ledley, "Practical problems in the use of computers in medical diagnosis," Proc. IEEE, vol. 57, November 1969.

[19] G. Mandler, "Organization and memory," in "The Psychology of learning and motivation," vol. 1, pp. 307-372, Academic Press, New York 1967.

[20] C. Hewitt, "Procedural embedding of knowledge in PLANNER," Proc. 2nd International Joint Conference on Artificial Intelligence, pp. 167-184, 1971.

[21] R. Simmons, "Natural language question answering systems," CACM, Vol. 13, No. 1, pp. 15-30, 1970.

[22] J. Rulifson, J. Derksen, R. Waldinger, "QA4: A procedural calculus for intuitive reasoning," Technical Note 73, SRI, November 1972.

SUMMARY

LOCAL LOGICS†

Richard Bellman
University of Southern California
USA

We may regard logics as a synthesis of experience. What is common sense in one world would be fantasy in another, and conversely. It was recognized about 2500 years ago that axioms were necessary to provide a firm mathematical foundation for geometry. This was a bold and imaginative step. This method was widely imitated. We may cite Newton, The Declaration of Independence, and the founders of modern probability theory.

The burden is on the user. He must show that the axioms are a good approximation of reality. Not much has been done by mathematicians on these problems of approximation.

The universe is strange. As has been pointed out, not only is the universe stranger than we think, it may be stranger than we can think. One result is that the physicist, who must deal with the real world, has at least three logics; one for ordinary state variables, classical mechanics, one for the microscope, quantum mechanics, and one for the macroscopic, relativity theory.

Thus, consistency gives way to utility. It is possible that one theory would handle all three cases, but it would probably be too complex to use.

We definitely need more logics to deal with uncertainty. The classical theory of probability is essentially a frequency theory. It works well in certain cases but works poorly in many others. The recent theory of Lotfi Zadeh, fuzzy systems, seems quite promising.

There are many interesting consequences as far as the use of computers and artificial intelligence are concerned. Human intelligence will always be required. Computers will become more powerful and the use of interactive computers will increase, but computers will always be an adjunct to the mind.

† This paper will appear in the book Modern Uses of Multiple-Value Logic.

FUZZY MODAL LOGIC[1]

Peter K. Schotch
Dalhousie University
Canada

I. The literature of many-valued modal semantics is concerned, almost entirely, with treatments of modal operators from the truth-functional point of view.[2] Perhaps the popularity of this sort of approach should come as no surpirse, since it has proved possible to recast many of these treatments in terms of possible worlds semantics.[3] The purpose of this study is to formalize certain types of non-truth-functional modal semantics using the notion of a fuzzy set.[4] In some ways this is a perverse enterprise since many proponents of many-valued semantics take the truth-functionality of their approach to be a definite virtue.[5]

When considering ways of fusing standard relational semantics with the 'fuzzy' viewpoint, two main approaches suggest themselves:

(1) the use of fuzzy valuations

(2) the use of fuzzy relations.

To elucidate these, we take standard (relational) modal semantics to be based on the notion of a relational model. This consists of two ingredients. The first is a frame $F = (\underline{W},R)$, where $\underline{W}$ is a non-empty set (intuitively, the set of possible worlds), and $R \subseteq \underline{W} \times \underline{W}$, is a binary relation (intuitively, the relation of relative possibility). The second is a valuation, $V:Nat \rightarrow 2^{\underline{W}}$ which is a function from the natural numbers to subsets of $\underline{W}$. Intuitively, V assigns to each atomic sentence P_n, the set of worlds in which P_n is true. Any valuation V, extends to all sentences in the following way:

(i) $V(\neg A) = W - V(A)$

(ii) $V(A \,\&\, B) = V(A) \cap V(B)$

(iii) $V(\Diamond A) = \{x \in W: \; xRy \wedge y \in V(A)\}$

From these we may derive the definition of $V(\Box A)$ as follows:

$V(\Box A) = \{x \in W: \; xRy \Rightarrow y \in V(A)\}$ using $\Box A = df \neg\Diamond\neg A$.

We then say that the wff A is <u>logically true</u> iff $V(A) = \underline{W}$ in all models, and in terms of this we may proceed to give a semantical account of many of the well-known systems of modal propositional logic.

II. Fuzzy valuations.

By a fuzzy model of the first kind, we understand a frame $F = (\underline{W},R)$ together with a <u>Zadeh valuation</u> $V_Z:Nat \rightarrow J^{\underline{W}}$. The latter is a function which assigns to each atomic sentence P_n a fuzzy characteristic function: C_{P_n} which takes on values from the closed unit interval. Intuitively, we may think of $C_{P_n}(x)$, as the degree to which

P_n is true in the world x, with $C_{P_n}(x) = 1$ meaning P_n is completely true in x, and $C_{P_n}(x) = 0$ meaning P_n is completely false in x.

V_Z may be extended to cover all wffs in several different ways, depending upon which sort of many-valued logic has the most appeal for us. We might choose, e.g. to adopt as our account of the truth-functional operators, the language FUZ which formalizes the sort of semantics proposed by Lakoff.[6] This requires our taking as the primitives the set: $\{\neg, \&, \rightarrow\}$ and putting the following condition on V_Z:

(i) $V_Z(\neg A) = 1 - V_Z(A)$

(ii) $V_Z(A \,\&\, B) = \text{Min}(V_Z(A), V_Z(B))$

(iii) $V_Z(A \rightarrow B)$ = the (two-valued) characteristic function of the set: $\{x \in \underline{W}: C_A(x) \leq C_B(x)\}$

On account of this last definition, $A \rightarrow B$ is always either completely true or completely false in every world x.

By taking the above line, or indeed any line in which we have only a single 'designated' value and deal with our operators in a truth functional manner, we commit ourselves to a non-classical view. In the present case, for example, if we take as our definition of logical truth (as seems natural) complete truth in all worlds (in all models) it turns out that $\neg(A \,\&\, \neg A)$ is not logically true.

As far as the non-truth-functional operators are concerned we, once again, have our choice. In order to stay as close as possible to the usual way of doing things, we shall adopt the following:

(iv) $V_Z(\Diamond A)$ = the (two-valued) characteristic function of the following set: $\{x \in \underline{W}: xRy \wedge C_A(y) \neq 0\}$

This makes $V_Z(\Box A)$ the two-valued characteristic function of the set:

$$\{x \in \underline{W}: xRy \Rightarrow C_A(y) = 1\}$$

Here again we make $\Diamond A$ either completely true or completely false in every world. This does not seem entirely out of accord with our intuitions however. In fact, we often use words like 'possibly' in ordinary discourse in order to make some definite statement in the face of uncertainty. We may not be sure about whether or not it is true to say that P, while at the same time being certain that at least it is true to say that possibly P.

In view of what was said above, the way our set of logical truths turn out is not entirely surprising. Suppose, for example, that we put no particular restriction on R. It is an easy matter to verify that in every model at no world does the value of $\Box(A \rightarrow B)$ ever exceed the value of: $\Box A \rightarrow \Box B$. Hence we have as a basic logical truth: $\Box(A \rightarrow B) \rightarrow (\Box A \rightarrow \Box B)$. Also, it is clear that we have as an admissible rule: From $\models A$, infer $\models \Box A$, thus for R simply binary we get a sort of non-classical version of K.[7] Similarly, for R reflexive we get as a logical truth:

$\Box A \rightarrow A$, for R reflexive and transitive: $\Box A \rightarrow \Box\Box A$, and for R reflexive and symmetric: $A \rightarrow \Box\Diamond A$.

III. Fuzzy relations.

If instead of the same old story of the elementary modal concepts, we want genuine fuzzy modalities, a distinctly new tack will have to be taken. In what follows, we preserve the classical account of the truth-functional operators while at the same time making the notion of relative possibility a matter of degree.

By a fuzzy model of the second kind, we understand a fuzzy frame $\mathcal{F} = (\underline{W}, \delta)$, with $\underline{W}$ as before and $\delta : \underline{W} \times \underline{W} \rightarrow J$ a two-place function, together with a valuation $V: \text{Nat} \rightarrow 2^{\underline{W}}$ which satisfies conditions (i) and (ii) of I. For this kind of frame we may interpret $\delta(x,y)$ as the degree to which the world y is possible relative to the world x.

To extend V to cover all wffs we add the following stipulation:

(iii') $V(\Diamond A) = \{x \in \underline{W}: \delta(x,y) = 1 \wedge y \in V(A)\}$

We can now formalize the semantics of the usual systems by adopting various conditions on δ. The difference between this approach and the standard one is that there is now a variety of ways, in many cases, to produce the classical (modal) systems. If we want T we must have that $\delta(x,x) = 1$ for every $x \in \underline{W}$, but to get S4 we could add to this any one of several stipulations. For example, we could add:

$$\delta(x,z) = \underset{y \in YZ_x}{\text{Max}} (\delta(x,y)) \cdot \underset{z \in YZ_x}{\text{Max}} (\delta(y,z))$$

where $YZ_x = \{y \in \underline{W}, z \in \underline{W}: \quad \delta(x,y) \neq 0 \wedge \delta(y,z) \neq 0\}$

or alternatively we could replace '=' in this by '$\geq$'.

It is easy to see that similar remarks apply to the case in which we want $\delta(x,y)$ to do the same job as a symmetric R.

The importance of this new freedom becomes clear when we decide to make use of the fuzziness of R in order to introduce the semantics of another modal operator. We add to our primitives, the unary operator M, and make the further stipulation on V:

(iv) $V(MA) = \{x \in \underline{W}: \delta(x,y) \neq 0 \wedge y \in V(A)\}$

The expression 'MA' can be read: 'it might be possible that 'A'. This is an expression which is often heard in ordinary speech, and one has the strong intuition that in most cases those who utter it think of themselves as asserting something distinct from: 'it is possible that it is possible A', since it seems that iterations of modal terms do not occur very frequently in ordinary discourse. It is this fact which is probably responsible for the view on the part of some philosophers that the system which best captures the ordinary view of possibility and necessity is S5. The two expressions are certainly distinct according to fuzzy models of the second kind. One has as a basic logical truth that: $\Diamond A \supset MA$, without the converse

also being logically true. It is also clear that: $MA \supset \Diamond A$ fails to be logically true in general, although we shall get the converse of this in any model which satisfies the condition: $((\delta(x,y) \neq 0 \wedge \delta(y,z) \neq 0) \Rightarrow \delta(x,z) \neq 0)$ (In fact we get $\Diamond\Diamond\ldots\Diamond A \supset MA$). This is because the truth of $\Diamond^n A$ at a world x_o means there exists a chain of worlds $x_1, x_2, \ldots x_n$ such that $\delta(x_i, x_{i+1}) = 1$ and $x_n \in V(A)$. But the condition gives $\delta(x_o, x_n) \neq 0$ and hence $x_o \in V(MA)$.

Depending upon which conditions we adopt for δ, we can make our account of M either close to, or remote from, our account of $\Diamond$. Let us say, for example, that we require $\Diamond$ to be as specified by T. This means that we must have $\delta(x,x) = 1$ for all $x \in \underline{W}$, as noted above. If we add the further specification on δ that:

If $\delta(x,y)$, $\delta(x,z) \neq 0$ then $\delta(x,z) \neq 0$ and $\delta(x,z) < \delta(x,y) \cdot \delta(y,z)$

If $\delta(x,y) \cdot \delta(y,z) = 0$ then $\delta(x,z) = 0$

(a sort of decreasing relative possiblity condition) then the account which we give of M, will be of the S4 type. If we add to the above:

$$\delta(x,y) \neq 0 \Rightarrow [\delta(y,x) \neq 0 \wedge \delta(y,x) < \delta(x,y)]$$

we shall have an S5 type of modal operator. Note that we have the obvious relation between $\Box$ and $\neg M \neg$, i.e. in general: $\neg M \neg A \supset \Box A$ is a logical truth. As an intuitive reading for '$\neg M \neg A$' one would have something like: 'it must be necessary that A'.

Although $\delta(x,y)$ was interpreted above as the degree to which y is possible relative to x, there are clearly other interpretations which are of interest. Among the most prominent of these are the probabalistic ones. We could decide, for example, to say that $\delta(x,y)$ is the probability that the world y is realized given that the world x is realized, a sort of transition probability. I say "sort of" because I wish to retain stipulation (iii') as the truth condition for $\Diamond$. This means that the conditional probability should be taken as the probability that y will be realized at some time or other, (as opposed to realized next), given that x is realized. Understood in this way, (iii') gives us a recognizable possibility operator, one which bears an obvious resemblance to the Diadorean concept of possibility.

If we choose the probabilistic way of looking at δ, then it is clear that we should insist that the reflexivity condition hold viz. $\delta(x,x) = 1$ for all $x \in \underline{W}$. Exactly which other conditions hold depends upon which sort of formal enterprise we have in mind. If we wish to make contact with recent work in the logic of physical theory (especially quantum mechanics) we may require also the symmetry condition: $\delta(x,y) = \delta(y,x)$ for all x,y. With this much we find that the relation '$\perp$' defined: $x \perp y$ iff $\delta(x,y) = 0$ is symmetric and irreflexive i.e. an orthogonality relation. This may be used to construct an orthologic[8] as a proper part of our system.

IV. Baroque models.

It is not clear that much interest surrounds the sort of semantics which results from a consideration of fuzzy models of both the first and second kinds. There does seem to be some application of both kinds of fuzziness, however, in the study of models allowing both classical and non-classical worlds.[9] We accommodate this idea via the concept of a <u>baroque model</u>. By this term we understand an n-tuple:

$$\langle (\underline{W}, \underline{Q}_1, \ldots, \underline{Q}_k, \delta), V, V_{\underline{Q}_1}, \ldots, V_{\underline{Q}_k}, \mathcal{V} \rangle$$

Where $\underline{W}$ is the set of classical worlds, and each $\underline{Q}_i$ is a set of non-classical worlds such that each of these sets are pairwise disjoint. δ is the two-place function introduced above except that in the present case δ is defined on: $\underline{W} \cup \underline{Q}_1 \cup \ldots \cup \underline{Q}_k$. V is a valuation satisfying (1), (ii) of I, and (iii') of III, and each $V_{\underline{Q}_i}$ is a Zadeh valuation: Nat $\rightarrow J^{\underline{Q}_i}$.

For the sake of simplicity, we shall consider only baroque models in which there is a single set, denoted by $\underline{Q}$, of non-classical worlds. $\mathcal{V}$, the baroque valuation, we now define for a language having as primitives: Imp, &, $\neg$, $\Diamond$, M

$\mathcal{V}(p_n) = \langle V(n), V_{\underline{Q}}(n) \rangle$

$\mathcal{V}(A \,\&\, B) = \langle V(A \,\&\, B), V_{\underline{Q}}(A \,\&\, B) \rangle$

$\mathcal{V}(\neg A) = \langle V(\neg A), V_{\underline{Q}}(\neg A) \rangle$

$\mathcal{V}(A \text{ Imp } B) = \langle V(A \supset B), V_{\underline{Q}}(A \text{ Imp } B) \rangle$

$\mathcal{V}(\Diamond A) = \langle V(\Diamond A), V_{\underline{Q}}(\Diamond A) \rangle$

$\mathcal{V}(MA) = \langle V(MA), V_{\underline{Q}}(MA) \rangle$

$\mathcal{V}$ will be well-defined as soon as we make the following stipulations: First,

$V(MA) = \{ x \in \underline{W}: \ \delta(x,y) \neq 0 \wedge (y \in \underline{W} \Rightarrow y \in V(A)) \wedge (y \in \underline{Q} \Rightarrow C_A(y) \neq 0) \}$

Next, $V_{\underline{Q}}$ satisfies the four conditions of section II with the following changes:

(a) $\underline{W}$ is replaced everywhere by $\underline{Q}$

(b) $A \rightarrow B$ is replaced in condition (iii) by A Imp B

(c) xRy is replaced in condition (iv) by $\delta(x,y) = 1$

together with the condition:

(v) $V_{\underline{Q}}(MA)$ = the characteristic function (two-valued) of the set:

$\{ x \in \underline{Q}: \ \delta(x,y) \neq 0 \wedge C_A(y) \neq 0 \}$

Finally, we make the following stipulation about $\delta(x,y)$:

If $x \in \underline{Q}$ and $y \in \underline{W}$, then $\delta(x,y) = 0$

If $x \in \underline{W}$ and $y \in \underline{Q}$, then $\delta(x,y) < 1$.

Taking for our definnition of $\models A$ (A is logically true): $\mathcal{V}(A) = \langle \underline{W}, V_{\underline{Q}}(A) \rangle$ now gives us several interesting results. The first is that there is now a much more dramatic difference between M and $\Diamond$ than we could achieve in the last section. One feature of this difference is that M is no longer a normal[10] modal operator, i.e. we can no longer infer $\models \neg M \neg A$ from $\models A$. This is because the non-classical treatment (the FUZ treatment) of the truth-functional operators in $\underline{Q}$ is allowed to affect the way M-expressions are evaluated in $\underline{W}$. This is by no means the most curious feature of $\neg M \neg$.

We notice also that because of the difference between Imp in $\underline{W}$ (according to V) and Imp in $\underline{Q}$ (according to V_Q) we also lose the rule of substitution of equivalents. That is we see that there are wffs, A,B such that while (A Imp B) & (B Imp A) (which we abbreviate: A Bimp B) is a logical truth, $\neg M \neg A$ Bimp $\neg M \neg B$ is not. Thus, in addition to being non-normal $\neg M \neg$ is also non-classical.[11] As opposed to the usual treatment of non-classical worlds, this way of allowing such items into our model enables us to have our cake, while eating at least part of it.[12] Thus only M (and of course operators defined by means of M) is affected by Q, $\Diamond$ continues to be a normal operator. Another point to notice is that we still have as a logical truth: $\Diamond A$ Imp MA (as well as: $\neg M \neg A$ Imp $\Box A$).

In addition we see that we do not completely lose the rule of necessitation for $\neg M \neg$. The rule will still work for at least those classical tautologies which are logically true according to FUZ. These include such things as e.g. A Imp A. Some interesting failures include: (A & $\neg$ A) Imp B. This comes out (completely) false in $x \in \underline{Q}$ for which $C_A(x) = \frac{1}{2}$ while $C_B(x) < \frac{1}{2}$. Thus, that a contradiction implies anything need not be necessary (in the sense that it is false that if must be necessary). Lest anyone think that $\neg M \neg$ (A Imp B) is 'paradox free' we note here that: $\neg M \neg$ (A Imp (B Imp A)) is logically true in the present scheme, (although the companion to this viz: $\neg M \neg$ ($\neg$A Imp (A Imp B)) is not).

Depending upon what we want to get rid of, we can imagine switching from a FUZ description of what goes on in $\underline{Q}$ to say, a Lukasiewiczian one. Or if we prefer we may use both, employing the general characterization of a baroque model and having as many $\underline{Q}$'s as we feel we need.

FOOTNOTES

[1] I apologise to Zadeh and his co-workers for the title of this paper which is misleading. Strictly speaking since the paper does not employ fuzzy modal truth-values it is not about fuzzy modal logic but rather about applications of fuzzy set theory to modal logic.

[2] My use of 'truth-functional' here is essentially that of [Cresswell b] p.25.

[3] E.G. See [Scott] who carries out this sort of project for many-valued logics of the Lukasiewicz type.

[4] In this connection see e.g. [Goguen], [Lakoff], [Zadeh].

[5] This is especially true of Lukasiewicz. See [Lukasiewicz].

[6] My account of this is taken from [van Fraassen].

[7] For a thorough treatment of the classical modal logics see [Hughes and Cresswell].

[8] For a recent treatment see [Goldblatt].

[9] These were introduced in [Kripke] in order to formalize the semantics of S2 and S3 as well as other non-normal logics. In that study they have the property that if x is non-normal x V (A) for all wffs A. Non-classical worlds also appear in [Cresswell] where they are used in much the same way as in the present study.

[10] This terminology is employed in [Kripke].

[11] This terminology comes from [Segerberg].

[12] In [Cresswell] non-classical worlds are used with a much broader definition of valuation, which results in a logic of a much more spartan nature.

BIBLIOGRAPHY

[Cresswell a] Cresswell, M.J.: "Intensional logics and logical truth", The Journal of Philosophical Logic, V. I (1972) pp.2-15.

[Cresswell b] Cresswell, M.J.: Logics and Languages, Methuen & Co., London (1973).

[Goguen] Goguen, J.A.: "The logic of inexact concepts", Synthese (1971).

[Goldblatt] Goldblatt, R.I.: "Semantic analysis of orthologic", Journal of Philosophical Logic, V. 3, (1974), pp.19-36.

[Hughes and Cresswell] Hughes, G.E. and Cresswell, M.J.: Introduction to Modal Logic, 2nd ed., Methuen, London (1972).

[Kripke] Kripke, S.A.: "Semantical analysis of modal logic II non-normal modal propositional calculi", Addison, Henkin, Tarski (eds.), in The Theory of Models, North Holland Publishing Co., Amsterdam (1965) pp.206-220.

[Lakoff] Lakoff, G.: "Hedges: A study in meaning criteria and the logic of fuzzy concepts", The Journal of Philosophical Logic, V. 2 (1973) pp.458-508.

[Lukasiewicz] Lukasiewicz, J.: "A System of Modal Logic", The Journal of Computing Systems, V. 1 (1953) pp.111-149.

[Scott] Scott, D.: "Background to formalization", in Leblanc (ed.) Truth, Syntax and Modality, North Holland Publishing Co., Amsterdam (1973) pp.244-273.

[Segerberg] Segerberg, K.: An Essay in Classical Modal Logic, Uppsala (1971).

[van Fraassen] van Fraassen, B.: "Lakoff's fuzzy propositional logic", mimeo'd. (1973).

[Zadeh] Zadeh, L.: "Fuzzy sets", Information and Control V. 8 (1965) pp.338-53.

POSSIBLE AUTOMATA

B.R. Gaines[*]
[*]Man-Machine Systems Laboratory,
Dept. of Electrical Engineering Science,
University of Essex, Colchester, U.K.

L. Kohout[*†]
[†]University College Hospital
Medical School,
University of London, U.K.

Abstract This paper is concerned with the widest class of automaton structures whose semantics is compatible with our notions of state and automaton. It is first shown that the conventional spectrum of deterministic, stochastic, fuzzy and non-deterministic can be fitted into a single framework which is complete. In particular new results on the normalization of fuzzy automata and the relationship between fuzzy and stochastic automata are derived. A practical counter-example is then developed that does not fit into this spectrum, showing it to be inadequate. This is based on the richer interpretation of the notion of possibility that is required in the analysis of system stability and reliability. Finally, basic arguments are advanced to show that the structure must be at least an ordered semiring and at most a commutative ordered semiring.

1 Introduction

The concept of an automaton, or state-determined machine, has come to play a substantial role in many disparate branches of science and engineering. The joint origins of the concept in biology and computer engineering have been succintly reviewed recently by Burks [1]. In a related survey Arbib [2] criticizes the applicability of current automata theory and suggests that many new developments and extensions are required. This criticism will be echoed by those who have recognized the concepts of automata theory as relevant to their own disciplines but have been disappointed in the dearth of applicable results.

The convictions, on the one hand, that the basic concepts of automata theory are relevant but, on the other, that the present developments are not sufficiently fruitful have prompted several workers to investigate new automaton structures, e.g. Arbib's tolerance automata [3] and Zadeh's fuzzy automata [4]. Through the very diversity of interests involved automata theory has grown up piecemeal with a variety of automaton structures and semantic interpretations. The continuing intermittent addition of new structures reinforces the impression that not just the development of the subject but perhaps also its foundations are, in some sense, incomplete.

This paper was motivated by our own experience in applying algebraic system theory to problems of system identification, stability and control, where we have found it necessary to define automaton structures that do not fit the conventional spectrum of deterministic, stochastic, fuzzy and non-deterministic automata. These new structures initially appeared to be representable as automata over modal logics rather than Boolean algebra. However, the need soon became apparent for mixed logics involving continuous probability intervals as well as discrete modalities, and the variety of possibilities led us to look for some more general approach.

It has been shown by Santos and Wee [5] that the main spectrum of deterministic, stochastic, fuzzy and non-deterministic automata can be fitted into a single formalism, but this is descriptive rather than axiomatic. It leaves open many questions: whether further automaton structures can be invented ad infinitum; what is the most general formulation; and so on. The search for generality is itself dubious unless backed by definite practical requirements expressed as semantic constraints. In this paper we take three distinct approaches to the problem of establishing the most general structure possible for an automaton: analysing first a sense in which the conventional spectrum of automata is already complete; secondly arguing from practical application requirements that this spectrum is inadequate; and thirdly, reversing the direction of increasing generality, to show by foundational arguments that certain quite powerful structural constraints are necessary to an acceptable concept of an automaton, i.e. that arbitrary algebraic structures formally similar to automata do not necessarily possess viable semantics.

The title of this paper is something of a play on words since we are concerned both with the weakest algebraic structures that are possible for automata, but have also chosen to develop all our exemplars within a range concerned with the expression of uncertainty, notably the fine, but important, distinction between the possible and the probable. Section 2 analyses the conventional spectrum of automata in terms of the underlying truth sets and semiring operators, and shows that it may be seen as a complete set of variants of two basic parameters. Section 3 analyses the requirements upon practical explicata of uncertainty about behaviour, and shows that they cannot be met by these variants. It develops an exemplary class of automata over a multi-valued logic of possibility and probability. The final section summarizes the extensions made and raises the converse question of the weakest possible structure that supports our concept of an automaton.

2 Conventional Automaton Structures

2.1 V-sets and Normalization

We take as our informal concept of an automaton at this stage the usual one of a machine with internal states and external inputs whose next state is a function only of its current state and current input. It is the form of this next state function, NSF, which is primarily of concern in this paper. It is only when a deterministic automaton commences in a known state that the next state itself is sharp, i.e. uniquely defined. Usually we have a statement about the current state of the automaton and the NSF enables us to infer a further statement about the next stage.

The standard forms of statement can all be represented as mappings from the set of states, S, to a truth set, V, $\delta: S \to V$; Goguen [6] calls such a mapping a V-set with S as carrier. For the purposes of describing automata states we also require normalization conditions expressing that the automaton is actually in one and only one state. We shall later take V to be a semiring with binary operations, $\oplus$ (we do not use '+' because it can be confused with arithmetic +) which is associative and commutative, and $\odot$ which is associative, often commutative, and distributes over $\oplus$. Hence it is convenient to express the normalization condition in terms of the formal expression $\bigoplus_S \delta(S)$, meaning the result of operating over the co-domain of δ in the truth-set with $\oplus$ (i.e. 'summation' over the truth set if $\oplus$ is actually +. We assume such summation is well-defined if S should be infinite).

By suitable choice of V and $\oplus$ we will show that we can take the normalization for the four cases considered to be:

$$\bigoplus_S \delta(S) = 1 \qquad (1)$$

The standard forms of statement are briefly reviewed in the following sections.

2.2 Deterministic States

These express the conditions that arise when a system's behaviour is completely defined and determinate. The automaton representing it is always in a well-defined, 'sharp', state. We can express this: for each state, it is true or false that the automaton is in the state and the automaton is in precisely one state. A suitable truth set is binary, $V \equiv \{0,1\}$, with $\oplus$ being arithmetic +, and the normalization as in equn.1. This necessitates only one state being mapped onto 1, and hence we could express the normalization as, 'the inverse image of 1 under δ contains just one element'.

2.3 Stochastic States

These express the conditions that arise when a system's behaviour is a Markov process whose behaviour is constrained by well-defined probabilities. The probability of the automaton representing it being in a particular state is then always well-defined. That is, for each state, the probability that the automaton is in the state is defined and the automaton is in precisely one state (the probabilities over all states sum to one and the conditional probabilities of the automaton being in one state given that it is in another are all zero). A suitable truth set is a closed interval of reals, $V \equiv [0,1]$ say, with $\oplus$ being arithmetic +, and the normalization as in equn.1.

2.4 Fuzzy States

Zadeh's concepts of fuzzy logic [7] and fuzzy automata [4] represent an attempt to provide a calculus of approximate reasoning. Formally, a fuzzy logic is a Lukasiewicz $\mathbf{L}_{N_1}$ system [8 p.337] but Zadeh [9,10,11] has contributed detailed semantics which make the application of the logic attractive and practically useful, for example in pattern recognition [12] and control engineering [13,14].

Hence fuzzy states express the conditions that arise when a system's behaviour is being described by a process of approximate reasoning. The degree of membership of a particular state of the automaton representing it to being the actual state is defined. If we take the usual fuzzy logic system with the truth set being the closed interval of reals and $\oplus$ being a MAX operator, then $V \equiv [0,1]$ and $a \oplus b = \text{MAX}(a,b)$.

2.4.1 Normalization of Fuzzy States

The normalization of fuzzy state sets to express the condition that the automaton is actually in precisely one state requires special attention. The published semantics of fuzzy automata seem confused on this point. Wee and Fu [15] state that if a state has a degree of membership of unity then the automaton is definitely in the associated state. However, the converse is not true and it

is possible for the rules of fuzzy logic to generate a situation in which the degrees of membership of all states are zero except one which is not unity. This is so even if the total degree of membership is "normalized" as suggested in [15] in the same way as a stochastic automaton (arithmetic sum of degrees of membership being unity). It also leaves open the meaning of two distinct states each having a degree of membership function of unity - an important case since it corresponds to the classical non-deterministic automaton.

It seems better to place the emphasis on the degree of membership of a state being zero as implying that the automaton is not in the associated state. With the usual fuzzy logic definitions of V and $\oplus$ given in section 2.4, our normalization condition of equn.1 requires only that at least one state has a degree of membership of unity. This condition is consistent with the definition of $\oplus$ in fuzzy logic, whereas the proposed "normalization" of [15] introduces arithmetic +, an operator outside fuzzy logic. Neither normalization is consistent with a degree of membership of unity implying that the automaton is definitely in the associated state, and this needs replacement.

A similar problem arises with non-determinate automaton and is clearly a semantic one to be resolved in actual applications. The formal normalization condition proposed here retains consistency between fuzzy automata and the others. We would propose the interpretation that a fuzzy automaton is definitely in a state if the truth values of all the other states are zero. The normalization then implies that the truth value of the remaining state is unity - the converse is not true.

2.5 Non-deterministic States

These might more positively be called 'possibilistic' since they express the conditions that arise when a system's behaviour is such that only the possibility and impossibility of its being in a given state can be discriminated. That is, for each state either it is possible, or impossible, that the automaton is in the state and the automaton is in precisely one state (at least one state is possible, and if only one state is possible then the automaton is in that state). A suitable truth set is binary, $V \equiv \{0,1\}$, with $\oplus$ being Boolean 'OR' which also corresponds to the MAX operation over this truth set. The normalization of equn.1 implies that the inverse image of 1 under δ contains at least one element (as it also does for fuzzy states).

2.6 From States to Transitions

For the moment we shall take it for granted that the nature of transitions can be expressed in terms of the same truth set as that for the states themselves. For example, a 'stochastic automaton' is one with stochastic states and stochastic state transitions. We can express the NSF as a function, $\delta: S \times S \to V$, which satisfies the normalization condition:

$$s \in S, \qquad \bigoplus_S (\delta(S) \otimes \sigma(S,s)) = 1 \tag{2}$$

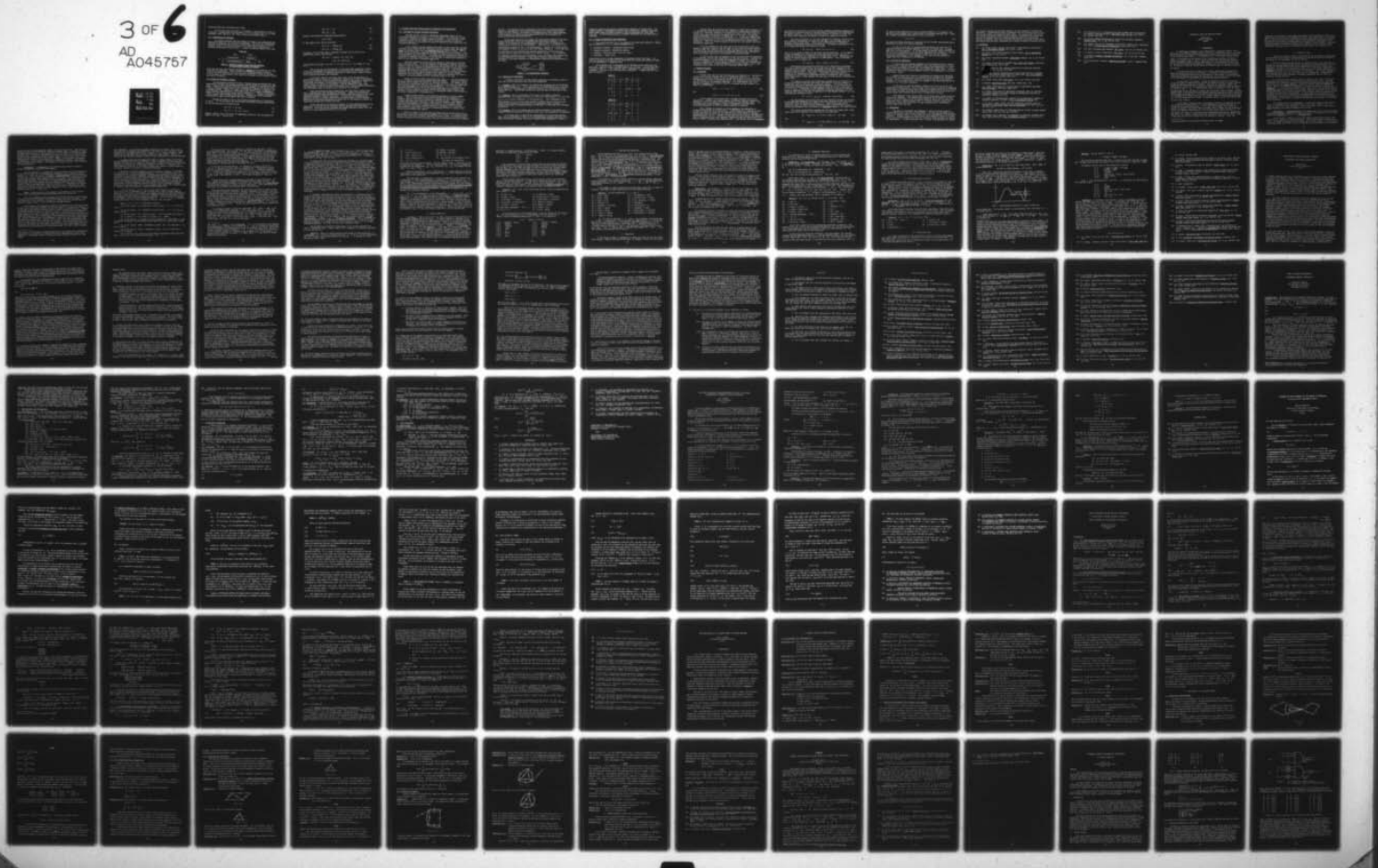
and where the V-set function, δ', after a transition is given by:

$$\delta': S \to V \equiv \bigoplus_{S'} (\delta(S') \otimes \sigma(S,S')) \tag{3}$$

Because $\otimes$ distributes over $\oplus$ we can show that the normalization of equn.1 is

AD-A045 757 INDIANA UNIV BLOOMINGTON DEPT OF COMPUTER SCIENCE F/G 9/2
PROCEEDINGS OF THE 1975 INTERNATIONAL SYMPOSIUM ON MULTIPLE-VAL--ETC(U)
MAY 75 G EPSTEIN N00014-75-C-0449
UNCLASSIFIED MVL-75-001 NL

3 OF 6
AD A045757



preserved under equn.3 provided equn.2 holds.

For the four cases discussed ⊙ is arithmetic x (multiplication) when ⊕ is arithmetic + (deterministic and stochastic), and ⊙ is Boolean 'AND', or the fuzzy equivalent 'MIN', when ⊕ is 'OR' ('MAX') (fuzzy and non-deterministic).

2.7 Comparisons and Contrasts

The previous sections have been phrased to bring out the similarities and differences between the four structures considered. Note that the normalization condition is uniformly that of equn.1, and the truth sets are either the entire interval, $[0,1]$, or its boundary points, $\{0,1\}$, whilst the transitions are uniformly represented by equns.3 and 2. A table of operators against truth sets:

	Truth set: $\{0,1\}$	Truth set: $[0,1]$	
⊕ = +	Deterministic	Stochastic	⊙ = x
⊕ = OR	Non-deterministic	Fuzzy	⊙ = AND

Table I Relations Between Truth Sets and Operators for the Standard Spectra of Automata

shows that the four cases analysed encompass a complete set of variations for these truth sets and operators. This is intuitively satisfying because it gives a closure over those automata which have been most extensively studied in the past. It is an answer in this context to the question of whether we can continually invent new forms of automaton.

2.7.1 Fuzzy and Stochastic Automata

The relationships expressed in Table I between fuzzy, non-deterministic and deterministic automata, and between stochastic and deterministic automata, are well-known. However, that between stochastic and fuzzy automata is less obvious and it is worth discussing whether this is just a mathematical formality or whether it has a semantic content. Clearly the common use of the interval $[0,1]$ corresponds to quite different interpretations of the values within it - a "degree of membership" appears as a far less precise concept than a "probability". Equally the operators, + and x, appear little related to MAX and MIN. However, the following argument demonstrates a closer correspondence than might be expected.

Consider two events, A and B, with respective probabilities of occurrence, p_A and p_B. If the two events are statistically independent then the probabilities of their conjunction and disjunction are:

$$p(A \wedge B) = p_A \times p_B \tag{4}$$

$$p(A \vee B) = p_A + p_B - p_A \times p_B \tag{5}$$

Suppose, however, that A and B are not independent events but that one implies the other, $A \rightarrow B$, say. Then we have:

$$p(A \wedge B) = p_A \tag{6}$$

$$p(A \vee B) = p_B \tag{7}$$

However, the direction of implication also gives us:

$$p(A) \leq p(B) \tag{8}$$

so that equns.6 and 7 may be re-written:

$$p(A \wedge B) = \text{MIN}(p_A, p_B) \tag{9}$$

$$p(A \vee B) = \text{MAX}(p_A, p_B) \tag{10}$$

Conversely, if the "fuzzy logic" conditions of equns.9 and 10 hold for two probabilistic events, then we have:

$$p(A \wedge B) = \text{MIN}(p(A \wedge B) + p(A \wedge \bar{B}), \; p(A \wedge B) + p(\bar{A} \wedge B)) \tag{11}$$

which implies that either $p(A \wedge \bar{B}) = 0$ or $p(\bar{A} \wedge B) = 0$, i.e. either $A \rightarrow B$ or $B \rightarrow A$.

Thus we see that the applicability of the fuzzy logic operations of equns. 9 and 10 to determining the probabilities of conjunction and disjunction of two probabilistic variables is completely equivalent to their being a logical relationship of implication between the variables.

In principle therefore the fuzzy logic rules of [4] are reducible to a probabilistic logic in which all variables are connected by a chain of implication. The converse condition to that generally found useful in application of probability theory where one attempts to make variables statistically independent. The "chain" concept is intuitively significant - the MIN operation in fuzzy logic expressing that a chain is as weak as its weakest link - the MAX operation expressing that alternative chains in parallel are as strong as the strongest.

These relationships between probabilistic and fuzzy logics indicate that Table I expresses more than mathematical formalism. Clearly the relationship demonstrated between fuzzy and probabilistic logics should also extend to the richer semantics developed by Zadeh in [9,10,11]. It would also be interesting for application studies to compare probabilistic and fuzzy logics in their relative efficacies for particular situations and relate this to the presence or absence of implications between the variables involved.

In the next section we develop an argument for state specifications that go beyond those of the four so far discussed, and for automata over mixed state structures. The final section discusses the greatest generality beyond which our notion of a state-determined machine will not carry.

3 Automata Over Multi-Valued Logics of Possibility and Probability

3.1 The Need for Further Automaton Structures

Although section 2.7 gives a satisfying completeness result for the conventional spectrum of automata, it in no way implies the sufficiency of these structures to represent all possible cases of interest. That they are in fact inadequate is best seen by example, and we shall give one which is itself of particular interest in the context of calculi of possibility and probability, and of multi-valued logics.

In our studies of system stability and control we have been very concerned to embody in our formulation the distinction between possible events that may not occur and possible events that are guaranteed to occur sooner or later. The former events correspond to problems that may arise and have to be avoided. They relate to regions of states which are reachable in terms of stability analysis but not reachable in terms of control. The second type of possible event, however, is responsive to feedback control since if the situation is continually recreated in which it may occur then it eventually will occur.

Note that probability theory does not provide an explicatum of the first type of possible event. If for the purposes of analysing an uncertain system we assign an uncertain event a non-zero probability then we imply that not only may it occur but also, in a sequence of occurrences each of which may be that event, it eventually will occur with a probability arbitrarily near one. The notional assignment of a definite probability to an event also fails to provide an adequate explicatum of the second type of possible event because it has the stronger implication that the relative frequency of such events in a sequence will tend to converge to the given probability with increasing length of sequence.

Either, or both of these connotations which proba[illegible]as over possibility may be too strong in practical situations where the concep[illegible] probability theory are being used to express the effects of uncertain behaviour. For example, we are often faced with situations where an event, E, may occur, but there is no guarantee that E actually will occur, no matter how long we wait. It we ascribe some arbitrary probability to E then we certainly express that it is a possible event. However we are in a position to derive totally unjustified results based on the certainty of some eventual occurrence of E, or meaningless numeric results based on the actual 'probability' of occurrence of E.

A similar problem arises in the practical application of linear systems theory. There are many results which may be derived from the assumption of linearity (such as the complete extension of knowledge of local behaviour to that of global behaviour) which are false in most practical systems. The engineer resolves these problems in practice by using a set of 'rules-of-thumb' based on commonsense and experience to constrain the deductions he is prepared to make. Such a resolution is however extremely difficult to implement in an automated, or computer-aided, design system and becomes increasingly difficult to apply as the system involved becomes more complex.

There is a danger on the one hand that results may be derived which have no justification other than an unwarranted strength in the theory. For example, Gaines [16,17] that a two-state stochastic automaton can solve a class of control problems otherwise requiring a recursive automaton [18] and not soluble by any finite automaton

[16,18]. This powerful result is dependent on the source of uncertain behaviour being truly probabilistic, and cannot be derived if it is merely possibilistic. There is no way, however, of preventing the consequences of this result appearing in the analysis of a system in which uncertainties have been represented by probabilities rather than possibilities.

On the other hand there is a danger that significant phenomena may be overlooked because they cannot be distinguished from such spurious effects. For example, it is possible to derive deterministic results about the behaviour of automata whose transitions are indeterminate. Starting in S_o, the states of the indeterminate automaton of Fig. 1 are indeterminate. However, it is clear that if the automaton is found to be in S_4 it must have passed through S_3. If we know that the transitions of this automaton are properly probabilistic then we may also conclude that its state will eventually be in the set (S_3,S_4). If, in addition, the transition probabilities are well-defined, we may also derive the expected time for this state set to be reached. These distinct forms of implication are confounded if we merely represent the indeterminate transitions as probabilistic in all cases.

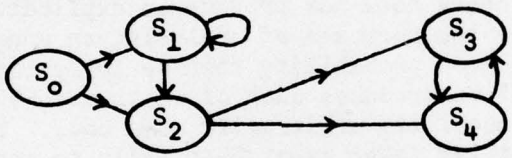


Figure 1 An indeterminate automaton

3.2 Explicata of Uncertainty

It appears that there are three distinct explicata of uncertainty, each of which has its own consequences that need clear separation:

(I) Possible Event E is possible - no reliance may be placed upon the occurrence or the non-occurrence of E. This corresponds to an interpretation of E as an event whose negative consequences must be taken into account, but whose positive consequences must not. Conventional probability theory provides no explicatum of this concept.

(II) Frequent Event E is frequent in the sense of the theory of infinite sequences, that in a sequence of events, E_i, for any N, there exists $M > N$, such that $E_M = E$, i.e. E occurs 'frequently' in the sequence E_i. This corresponds to the interpretation of E as an event whose eventual occurrence may be relied upon, but whose relative frequence of occurrence is not necessarily stable or known. A possible explicatum in probability theory is that $p(E) > 0$, the event if of non-zero probability.

(III) Probable Event E is frequent and its relative frequency of occurrence in a sequence of events converges to a definite value, p(E), its probability of occurrence.

It is clear that, in terms of our classification in the previous section, case I may be represented in terms of the non-determinate automata, and case III in terms of stochastic automata, with case II possibly being represented in terms of

stochastic automata with properly probabilistic transition of unknown value. We needed, however, to encompass the cases of mixed transitions, typically situations where certain types of possible behaviour could be guaranteed but other types could not. In the following section a suitable formulation for the mixed case is developed.

3.3 A Logic of Possibility and Probability

Let us take the truth set, V, to consist of the semi-open interval, $R \equiv (0,1]$, and the elements, N, F, P, I, whose interpretation is:

N - Necessary occurrence - probability equals unity.

F - Frequent occurrence - probability unknown.

P - Possible - cannot say that it will not occur.

I - Impossible - cannot occur.

A truth value in R is a known probability of occurrence which is not zero. We shall say an event is of type R if its truth value is in R and will write R:p, where p is its probability, to emphasize this.

The ⊕ operator over V corresponds to two different routes arriving at the same state - what can we say if we know either x or y is true. A truth table for ⊕ is given in table II. The ⊙ operator over V corresponds to a state followed by a transition - what can we say if we know that y follows x. A truth table for ⊙ is given in table III.

TABLE II

⊕	N	F	R:r	P	I
N	N	N	N	N	N
F	N	F	F	F	F
R:r'	N	F	R:r+r'	F	R:r'
P	N	F	F	P	P
I	N	F	R:	P	I

TABLE III

⊙	N	F	R:r	P	I
N	N	F	R:r	P	I
F	F	F	F	P	I
R:r'	R:r'	F	R:rr'	P	I
P	P	P	P	P	I
I	I	I	I	I	I

Consider first the structure with R taken as a single logic variable, i.e. $V \equiv \{N,F,R,P,I\}$, which allows for all the explicata of uncertainty developed in section 3.2 Note that, without R, the tables for ⊕ and ⊙ are simply those of a 4-value Post logic, and hence can be mapped onto a fuzzy logic. R, however, behaves anomalously in that R⊙P = F whereas R⊙P = P. It has been suggested by Brown that the V-set of a fuzzy logic be taken to be a distributive lattice [19]. However the interaction of R and P is inconsistent with ⊕ and ⊙ being lattice operations. This is a concrete example of the need for more general truth sets discussed by Goguen [6].

If we now consider the full truth set as first specified in which R is actually a semi-open interval, then the logic is now a mixed continuous discrete structure which can, however, still be neatly represented in the 'truth tables'. Such structures are both theoretically interesting and practically necessary to obtain rich enough explicata of the behaviour of uncertain systems.

It will be noted that the diagonals of the two tables show the idempotency of the elements, and the wider significance of this may be raised. However, the individual elements of R are clearly not idempotent in general ($p+p \neq p$, and $pxp \neq p$, in general), and if we consider a variant on F, such as G interpreted as 'properly probabilistic' (unknown probability in the open interval, (0,1)), then idempotency can be seen to fail even for a discrete element (G⊙G = F).

4 Possible Automata

4.1 Semirings

We have noted that the truth set need not be a fuzzy set or a distributive lattice, and that the elements need not be idempotents under ⊕ or ⊙. In the example of the previous section it can be seen that ⊕ and ⊙ are both associative and commutative and that ⊙ distributes over ⊕, i.e. together they give the truth set the structure of a commutative semiring. It is also apparent that this semiring is positive [19, p.125] in that if we consider the zero element (I in Tables II and III) then:

$$a \oplus b = I \rightarrow a = I = b \tag{12}$$

and:

$$a \odot b = I \rightarrow a = I \text{ or } b = I \tag{13}$$

In this example we have shown that a stronger structure would be too restrictive. However, the question remains of whether a positive commutative semiring is still too strong a structure on which to base automata theory. The following notes outline arguments to show on fundamental, and intuitively satisfying, grounds that at least an ordered semiring is necessary.

First consider the operator, ⊕, which represents the combination of different trajectories to the same state. Trajectories may be combined in pairs so that this gives the truth set the structure of a partial groupoid (partial because some pairs of values may not arise and hence their result is undefined, e.g. probabilities of 1 and 1). However, we must also take into account the independence of trajectories, that they represent alternative paths and there should be no effect of order or grouping when combining them. This implies that ⊕ is necessarily

commutative and associative, and hence defines a partial commutative semigroup over the truth set (it may be taken as a partial monoid by adding the null trajectory as an identity element). We may drop the term "partial" in general by noting that the "don't care" conditions can always be fitted in to complete the monoid.

Even these constraints do not fully represent the necessary structure since each trajectory terminating in a state can only add to our knowledge about the automaton being in that state. There can be no cancellation of information obtained by considering independent trajectories. One possible expression of this is to require the monoid to be positive, so that:

$$\forall \quad a,b \in V, \quad a \oplus b = 0 \rightarrow a = 0 = b \tag{14}$$

where 0 is the identity element of the monoid written additively. It can readily be seen (by adding a or b to each side of the left equn. of (14)) that if the elements of the monoid are idempotent (14) automatically holds. Idempotency also implies the natural order on the monoid is a semi-lattice. That is defining a relation, $\geq$, on V in terms of $\oplus$:

$$\forall \quad a,b \in V, \quad a \geq b \quad \leftrightarrow \exists\, c : a = b \oplus c \tag{15}$$

Unfortunately the positivity condition of (14) alone does not guarantee that this is even a partial order, and it seems that the best statement of the constraint upon the monoid is that the natural pre-order on it defined by (15) is actually a partial order. This itself implies that the monoid is positive and is implied if the elements are idempotent. Intuitively, this order relation corresponds to our having two independent sources of information about a state which cannot cancel - taken together they must give at least as much information as either alone.

The operator $\otimes$ presents more interesting problems since it represents the interaction between states and transitions, and there is no a priori reason to suppose that they can be expressed in the same language. Let us start with the more general assumption that the transitions are drawn from a set of functions, $F \equiv \{f: V \rightarrow V\}$. Considering the same argument as for $\oplus$, it can be seen that the result of applying a function to each individual trajectory separately (and then combining them) must be the same as applying it to them already combined - i.e. the functions must distribute over $\oplus$:

$$\forall \quad f \in F, \quad a,b \in V, \quad (a \oplus b)f = (af) \oplus (bf) \tag{16}$$

The implications of distributivity are not intuitively obvious and they may be expressed more meaningfully in terms of the order relation of (15), since (16) shows that f must be isotone with respect to $\geq$. Again we may argue that a transition cannot in itself increase information about a state so that f must be isotone non-increasing (this also makes it a residuated mapping in the sense of [21]).

The isotone non-increasing mappings over the truth set clearly form a semigroup which can be extended to be a semiring by the definitions:

$$\forall \quad f,g,h \in F, \quad h = f \otimes g \quad \leftrightarrow \forall a \in V, \quad ah = afg \tag{17}$$

and:

$$\forall \quad f,g,h \in F, \quad h = f \oplus g \quad \leftrightarrow \forall a \in V, \quad ah = af \oplus ag \tag{18}$$

The partial order defined by (15) has a natural extension to F in terms of (18) and this in turn implies that F under $\oplus$ and $\odot$ is a positive semiring [20]. There are a number of possible injections of V into F such that:

$$af = a \odot f \tag{19}$$

and hence the entire structure of the monoid and its relevant endomorphisms may be represented in terms of a positive semiring.

It will be noted that the examples given previously are such that $\odot$ is commutative whereas no informal arguments have been put forward here to suggest that this is true in general. It is easy enough to generate simple structures in which $\odot$ is not commutative but all our other requirements are satisfied. We have yet to find a semantics for such structure to show that they are necessary. Conversely there appears to be no argument on the lines of those advanced to suggest that such a semantics is not possible.

4.2 The Role of Idempotency

If one accepts the informal arguments of the previous section in terms of the monoid over V representing "information" about the automaton being in a state then it would be natural to assume that its elements were idempotents, i.e. that getting the "same" information a second time contributed nothing extra. Only the probabilistic case gives a counter-example, and here the "information" is a value rather than a datum.

Suppose however that instead of considering the probabilities themselves one considers the underlying Borel set structure of the σ-algebra for the probabilities. Then the "information" consists of disjoint sub-sets whose measures correspond to the probabilities and if $\oplus$ is regarded as the union operation on the sub-sets it is, of course, idempotent.

In this case our semiring becomes a lattice, as it was for all the non-probabilistic examples given. Thus it might well be that an intuitively satisfying axiomatization of automata theory could lead to the stronger structure of a lattice, rather than a semiring, providing one is prepared to carry the full structure of a measure algebra when carrying results over to probabilistic automata.

This suggestion throws further light on the relationship between fuzzy and probabilistic automata. The normalization conditions are the same in that the joins of the truth values for all the states should be units, but the fuzzy truth values must form a linearly-ordered chain (a "vertical" section), whereas the probabilistic truth values must form a totally unordered set (a "horizontal" section) whose meets are zero.

5 Conclusions

This paper is exploratory and intended to 'open up' certain aspects of automata theory and of the logic of uncertainty. We have been concerned to stay close to the semantic roots of these topics and avoid over-emphasis on mathematical formalism. Automata theory to a large extent, and probability theory to a lesser extent, have evolved pragmatically with new constructs being introduced to satisfy new requirements. It seems appropriate now to return to fundamentals and examine

the minimum underlying sub-structure common to all our concepts of automata and uncertainty. Goguen [6] has given an extremely clear and coherent account of the logic of uncertainty in category-theoretic terms. The present paper may be seen as a further exploration within the same framework, illustrating on one hand the need for the systems within that framework that go outside the conventional spectrum of automata, and on the other hand defining the boundaries of that framework beyond which the basic connotations of a structure being an automaton are lost.

6 References

[1] A.W. Burks, "Logic, biology and automata - some historical reflections", Int. J. Man-Machine Studies, vol. 7, pt. 3, 1975.

[2] M.A. Arbib, "From automata theory to brain theory", Int. J. Man-Machine Studies, vol. 7, pt. 3, 1975.

[3] M.A. Arbib, "Tolerance automata", Kybernetika (Prague), vol. 3, pp. 223-233, 1967.

[4] L.A. Zadeh, "Fuzzy sets and systems", Proc. Symp. Syst. Theory, Polytechnic Inst. Brooklyn, pp. 29-37, 1965.

[5] E.S. Santos and W.G. Wee, "General formulation of sequential machines", Inf. Contr., vol. 12, pp. 5-10, 1968.

[6] J. Goguen, "Concept representation in natural and artificial languages: axioms, extensions and applications for fuzzy sets", Int. J. Man Machine Studies, vol. 6, pp. 513-561, Sept. 1974.

[7] L.A. Zadeh, "Fuzzy sets", Inf. Contr., vol. 8, pp. 338-353, June 1965.

[8] N. Rescher, Many-Valued Logic, New York: McGraw Hill, 1969.

[9] L.A. Zadeh, "Fuzzy logic and its application to approximate reasoning", Proc. IFIP Congr., Stockholm, August 1974.

[10] L.A. Zadeh, "Fuzzy logic and approximate reasoning", Memo. No. ERL-M479, Electronics Research Laboratory, College of Engineering, University of California, Berkley, Nov. 1974.

[11] L.A. Zadeh, "A fuzzy-algorithmic approach to the definition of complex or imprecise concepts", to appear in Int. J. Man-Machine Studies 1975.

[12] P. Siy and C.S. Chen, "Fuzzy logic for handwritten numerical character recognition", IEEE Trans. Syst., Man and Cybernetics, vol. SMC-4, pp. 570-575, Nov. 1974.

[13] E.H. Mamdani, "Application of fuzzy algorithms for control of simple dynamic plant", Proc. IEE, vol. 121, pp. 1585-1588.

[14] E.H. Mamdani and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller", Int. J. Man-Machine Studies, vol. 7, Jan. 1975.

[15] W.G. Wee and K.S. Fu, "A formulation of fuzzy automata and its application as a model of learning systems", IEEE Trans. Syst. Sc. Cyb., vol. SSC-5, pp. 215-223, July 1969.

[16] B.R. Gaines, "Memory minimization in control with stochastic automata", Electron. Lett., vol. 7, pp. 710-711, 1971.

[17] B.R. Gaines, "The role of randomness in cybernetic systems", Proc. Cybernetics Soc. Conf. on Recent Topics in Cybernetics, London, Sept. 1974.

[18] E.M. Gold, "Universal goal seekers", Inf. Contr., vol. 18, pp. 395-403, 1971.

[19] J.G. Brown, "A note on fuzzy sets", Inf. Contr., vol. 18, pp. 32-39, 1971.

[20] A. Eilenberg, Automata, Languages and Machines, vol. A, New York: Academic Press 1974.

[21] T.S. Blyth and M.F. Janowitz, "Residuated mappings", Oxford: Pergamon Press 1972.

ŁUKASIEWICZ LOGIC AND FUZZY SET THEORY[1]

Robin Giles
Queen's University
Canada

1. Introduction

In developing a language for the formalization of physical theories [4,5] I have been led to abandon ordinary logic in favour of a nonclassical logic. This logic, defined by means of a dialogue interpretation, reduces with certain mild assumptions to the infinite-valued logic $Ł_\infty$ of Lukasiewicz [16]. (With much stronger assumptions - not acceptable in the context of the formalization problem - it reduces further to classical logic.) Now, it turns out that, with this dialogue interpretation, Łukasiewicz logic is exactly appropriate for the formulation of the "fuzzy set theory" first described by Zadeh [19]; indeed, it is not too much to claim that $Ł_\infty$ is related to fuzzy set theory exactly as classical logic is related to ordinary set theory.

It is my object here to justify this claim by developing the elements of fuzzy set theory in terms of the new logic. I first give the motivation for this logic, whose essential feature is a new definition of the concept of a proposition. I describe the dialogue interpretation and show how it gives rise to the truth tables of $Ł_\infty$ and leads to a new method for establishing logical identities.

We then pass to set theory. Instead of adopting the notion of a set as a primitive concept I assume that each set may be defined in terms of a unary predicate (property) that characterizes its members. When this is expressed formally the result may be read in two ways, according to whether the underlying logic is taken to be classical logic or $Ł_\infty$. (Syntactically, they are identical.) With the former interpretation we get ordinary set theory; with the latter fuzzy set theory. Similar remarks apply to the definitions and theorems characterizing various simple operations and relations involving sets. In the final section this approach is used in giving a brief treatment of convex fuzzy sets, and a simple theorem is stated and proved using $Ł_\infty$ and the dialogue method of proof. In general, however, proofs have beem omitted owing to lack of space.

Since Zadeh's original paper a considerable literature has appeared on fuzzy set theory and its applications. It is not appropriate to review this here but I should like to mention three items: first, the work of Goguen [8,9], which is similar in spirit to the present paper; secondly, that of Chang, who has developed in detail the algebraic properties of a class of many-valued logics including $Ł_\infty$ [2] and applied this work to the definition and metamathematical study of a generalized set theory [3]; thirdly, a neglected series of papers by Klaua [10-15], which I came across only when the work reported here was essentially complete. Klaua (see

[1]Research supported by the National Research Council of Canada.

especially [11]) develops a "many-valued set theory" based on Łukasiewicz logic in a manner similar to (but much more sophisticated than) that adopted here. In particular, most of the logical identities given here have been independently obtained by Klaua. His work goes far beyond the present paper in the mathematical development of the subject and deserves much more attention than it appears to have received. Both Klaua and Chang discuss, in different notations, the concepts of "bold" conjunction and disjunction introduced here in §4, and Klaua applies these to set theory as in §7.

2. A Pragmatic Logic

I shall first give the original motivation for the new approach to logic and then describe how it applies to those practical situations which have provided the motivation for the development of fuzzy set theory.

In order to give a precise account of a physical theory it is necessary to describe how the verifiable assertions of the theory are to be interpreted. For this purpose the language employed must include expressions which refer to particular experiments. In fact, it is sufficient to refer only to "elementary experiments", an elementary experiment being one which has only two possible outcomes, which we may designate "yes" and "no". For precision it is desirable to use a formal language for this purpose, and it turns out [4,5] that there is a natural way of doing this in which the expressions which designate elementary experiments have the syntactic structure of prime sentences.

At first sight this situation, which is familiar in the case of formalized mathematical theories, seems to accord perfectly with classical logic: one need merely describe a sentence as true if the corresponding elementary experiment gives the outcome "yes" and false otherwise. However, a serious difficulty immediately arises. Many of the elementary experiments referred to in physical theories are dispersive: i.e. they have the property that a repetition of the experiment does not necessarily produce a repetition of the outcome. This is most obvious in the case of quantum mechanics, but it occurs also with other theories, being then usually ascribed to thermal or statistical "fluctuations". To take account of this situation it is necessary either to give up interpreting the expressions mentioned above as prime propositions - which turns out to be very unsatisfactory - or to develop some alternative approach to logic which is capable of dealing with the situation. I follow the latter route here.

The changes required are fundamental. We must give up the classical definition of a proposition as "a statement that is either true or false", since the notion of truth is no longer available. What is to be done then? In the case of a prime proposition an obvious alternative is as follows:

Definition 1. A prime proposition is an expression to which there corresponds a definite elementary experiment. If this experiment is dispersive we call the prime proposition dispersive; if not, it is dispersion-free.

So far so good. However, to deal with compound propositions we need a new notion of the concept of a proposition itself. To this end we shall be quite realistic. We regard a proposition as a statement made by some speaker. It does not tell us anything about the "world"; at best it informs us about the beliefs of the speaker,

and in the case of an irresponsible speaker it may not even do that. What can we do to make the best of this situation? Clearly, we cannot expect a statement to represent more than the belief of the speaker, but we can at least try to avoid irresponsibility. Now, in practice irresponsibility is held in check in the following way. He who asserts a proposition incurs a certain risk, if only of loss of face, if his assertion should turn out to be unjustified. Indeed, in jurisprudence this aspect is emphasized: in many cases an assertion is considered to entail a legal commitment on the part of the speaker. This suggests a possible solution. Let us try the following definition:

Definition 2. A proposition is an expression whose assertion entails a definite commitment on the part of the speaker.

Thus the "meaning" of a proposition is now to be given not by describing conditions for its truth but by assigning some commitment that is assumed by him who asserts it. I shall call such an assignment a tangible meaning for the proposition. In order that a prime proposition, as defined above, should be a proposition in this sense we must associate some commitment with its assertion. Since the assertion should surely represent a belief that the corresponding elementary experiment will have outcome "yes" we may suppose that the commitment takes the form of a penalty should a trial of this experiment yield the outcome "no". Conceivably, the penalty could vary arbitrarily from case to case, but for the sake of simplicity we shall assume that it has the form of a constant (negative) utility, which we may for convenience denote by \$1.[2] This gives:

Rule 1. He who asserts a prime proposition undertakes to pay \$1 should a trial of the corresponding elementary experiment yield the outcome "no".

It is to be understood that when one trial has been carried out and the debt (if any) paid then the obligation incurred under Rule 1 has been discharged. Notice that repetition of the assertion increases the commitment and can thus be used to express a stronger belief - a fact well known to young children!

I now interrupt the argument to explain the connection with fuzzy set theory. In the above we have been concerned with the most exact of all sciences, physics. Let us now turn to the opposite extreme, the use of language in ordinary life. Consider an everyday proposition such as "John Smith is tall". Although we may perhaps agree that everyone over 7' and no one under 4' is tall, it clearly does not conform to normal usage to lay down any particular height and claim that everyone over and no one under that height is tall. Nevertheless, the proposition clearly has a useful meaning. For the clarification of this meaning it is natural to ask how we might decide - for instance to settle a bet - whether "John is tall", with the understanding that "tall" is used in the sense of the man in the street. The obvious answer is to ask the man in the street! We might agree, for instance, on the following procedure: let us go with John Smith into the street and ask the first man we meet whether he considers John tall; his opinion will be considered final. Now this is just an elemen-

---

[2]The notion of utility that is introduced here may be left vague, since everything we assume about it is embodied in a basic axiom that is described below (just before Theorem 1.) The present informal introduction merely provides motivation for that axiom.

tary experiment - a well-defined procedure resulting in an outcome, "yes" or "no". Thus with such an agreement the proposition, "John Smith is tall", becomes a "prime proposition" in the sense defined above. Moreover, it is clearly dispersive - repetition of the experiment will not always produce the same outcome. Thus we are faced with just the same situation as arises in the formalization of physical theories.

It seems likely that any proposition framed in common language can be given an exact meaning in the sense of Definition 1 (and in agreement with the intuitive meaning) in a similar way: namely, by laying down a definite procedure of arbitration, involving the selection of and subsequent appeal to an "umpire". Naturally, the umpire need not be the man in the street; perhaps more typically he may be some sort of "expert witness" or possibly a particular individual.

Returning to the main argument, we must now assign a meaning, in accordance with Definition 2, to every compound proposition. As in the classical case it is convenient to use a recursive procedure, explaining the significance of each logical connective by describing the commitment associated with any compound proposition in which it is the main connective in terms of those associated with its components. Of course, the choice of these commitments is a matter for deliberation, the aim being that (a) the meanings assigned to the logical connectives should be acceptable as explications of the corresponding intuitive notions, (b) the language so defined should be of practical value, and (c) this language should reduce (in a suitable sense) to classical logic when applied to dispersion-free propositions.

I now put forward a set of rules in which this procedure is carried out. These rules are to be understood in the following terms. When a speaker makes an assertion it is at first to be regarded as an offer to assume the indicated commitment. Any other speaker may respond to this offer, whereupon a debate ensues between these two speakers, in which no other speaker may intervene. The rules govern the conduct of this debate. (We use the usual logical symbols, $\wedge$, $\vee$, $\neg$, $\rightarrow$, $\forall$, $\exists$, denoting respectively "and", "or", "not", "implies", "for all", and "for some".)

Rule 2. Let P and Q be arbitrary sentences.

(a) He who asserts $P \vee Q$ undertakes to assert either P or Q at his own choice.

(b) He who asserts $P \wedge Q$ undertakes to assert either P or Q at his opponent's choice.

(c) He who asserts $P \rightarrow Q$ offers to assert Q if his opponent will assert P .

(d) To assert $\neg P$ is the same as to assert $P \rightarrow F$, where:

(e) He who asserts F promises to pay his opponent \$1. (The symbol F may be regarded as denoting a distinguished prime proposition which always gives outcome "no".)

(f) He who asserts $\exists x P(x)$ undertakes to assert $P(a)$ for some term a of his own choice.

(g) He who asserts $\forall x P(x)$ undertakes to assert $P(a)$ for some term a chosen by his opponent.

(In (f) and (g) $P(x)$ denotes an open sentence involving a single free variable x and $P(a)$ is the result of replacing x by a .)

The crucial rule is (c). To motivate it consider the case when P and Q are prime propositions. The assertion $P \rightarrow Q$ may then be interpreted as expressing a belief that P is at least as likely as Q to yield outcome "no", an interpretation which accords with classical material implication. In the general case the rule is to be understood in the following terms: If either speaker asserts a proposition of the form $P \rightarrow Q$ then the other speaker must either admit it (the assertion is then simply annulled) or challenge it by asserting P. In the latter case the first speaker must discharge his obligation by asserting Q, the original assertion then being annulled. An assertion may be challenged only once.

By (d) and (e) the assertion of $\neg P$ amounts to an offer to pay \$1 if a trial of P should yield "yes", which is a tangible way of expressing the belief that the outcome will always be "no". The definition of $\vee$ requires no comment. For $\wedge$ observe that although he who asserts $P \wedge Q$ is obliged to assert only one of P and Q he must be prepared to assert either, since he cannot tell which is opponent may choose. The rules for $\exists$ and $\forall$ are natural extensions of those for $\vee$ and $\wedge$.

Through these rules a tangible meaning is assigned to every proposition. In fact, suppose an arbitrary proposition is asserted by one speaker, say "me". The rules then govern an ensuing debate - with "you", say - which ends in a final position in which we are committed to a number of prime propositions. These propositions are then tried, and the debts incurred under Rules 1 and 2(e) paid.

With dispersive prime propositions the same final position will result in different payments on different occasions. However, *depending on my beliefs*, there will be certain final positions which seem "acceptable" to me, in that I expect on average a non-negative gain. To reach one of these is, as far as I am concerned, a "win". Thus I will be willing to assert a given proposition P iff (= if and only if) I am able to conduct the ensuing debate in such a way that an acceptable final position is assured. When this is the case we shall say P is true (for me) and write $\models P$.[3] If several speakers are under consideration we express the statement "P is true for a speaker S" by writing $S \models P$. In this way truth reenters the theory, albeit only in a subjective form.

As an example, suppose I assert the proposition $P \rightarrow Q$, where P and Q are prime. If you challenge by asserting P then I must reply with Q. The final position so reached may be acceptable to me - for instance if (in my opinion) the probability that P will yield outcome "no" is at least as great as that for Q - and in this case $P \rightarrow Q$ will be true for me: i.e. $\models P \rightarrow Q$. [Note that $\models Q \vee \neg P$ holds only if I am sure that Q will yield "yes" or that P will yield "no". Thus in this logic $P \rightarrow Q$ and $Q \vee \neg P$ are not equivalent.]

---

[3] The symbol "$\models$" belongs to the metalanguage: $S \models P$ is a proposition in the metalanguage expressing a property of the proposition P. Since we assume all propositions in the metalanguage are dispersion-free we can use classical logic there, and are not obliged to assign a tangible meaning to $S \models P$. (It is also possible to introduce $\models$ into the primary language, for instance by the following rule: he who asserts $\models P$ agrees either to pay \$1 or to assert P n times, where n is any positive integer chosen by his opponent.)

" P is false" will mean "$\neg$ P is true": i.e. $\models \neg P$. This is a much stronger statement than $\not\models P$, which denotes "it is not the case that P is true". For instance, a prime proposition P is true for me iff I am sure that a trial will yield outcome "yes", and false iff I am sure the outcome will be "no". A dispersive proposition is neither true nor false.

To develop the logic that should govern the reasoning of a rational speaker in these circumstances one must give some interpretation of "rationality" in terms of assumptions about the structure of the set A of acceptable final positions. The stronger these assumptions the simpler the resulting logic. As an extreme case we might assume that, for every prime proposition P , either the final position, denoted $\emptyset|P$, in which I have asserted P while you have asserted nothing or the position $P|F$ in which you have asserted P and I have asserted F is acceptable. (This amounts to saying that either P is true or $\neg P$ is true.) With the addition of certain other assumptions (described below) this leads to a logic which essentially coincides with classical logic. It has the same logical identities and differs only in respect of the interpretation of a proposition (Definition 2). However, in this sense of rationality no rational speaker would recognize any proposition as dispersive, and we have seen that this "classical" attitude is too narrow both for the formalization of physics and for everyday life. At the other extreme we might assume very little about the structure of the set A . This leads to a rich language, but to a relatively complicated logic that is not yet fully worked out. The complication is due to the fact that it is necessary to add to the rules of debate certain principles governing the order in which the various outstanding commitments should be discharged.

A reasonable compromise that admits the possibility of dispersive prime propositions but avoids these difficulties of order (in fact, it turns out that the "order of play" makes no difference) is given in [5] and [6]. The basic axiom may be stated in two parts. (Adopting only the first part leads to the logic just mentioned.) In the first part certain relatively uncontroversial assumptions are put forward: first, if α and β are two acceptable final positions then their "sum", namely that final position in which each speaker's commitment is given by adjoining his commitments in α and in β , is also acceptable; secondly, for any prime proposition P , the positions $F|P$ and $P|\emptyset$ are acceptable, while the position $\emptyset|F$ is not acceptable; thirdly, there is an Archimedian postulate which (roughly speaking) asserts that any position which I would be willing to accept on payment of an arbitrarily small fee is itself acceptable. The second part contains a postulate of a more arbitrary nature. For any final position α let $-\alpha$ denote the position obtained by exchanging the roles of the two speakers. Then the postulate asserts: for any final position α either α or $-\alpha$ (or both) is acceptable. (This corresponds to a familiar assumption in the foundations of Bayesian statistics.) If the set of final positions acceptable to a speaker satisfies this postulate I shall call the speaker probability-definite. The reason for this term is explained below.

The assumption of probability-definiteness is the crucial postulate that leads to Łukasiewicz logic. Indeed, using it (i.e. assuming "I" am probability-definite) one can show [6]:

Theorem 1: There is a unique function which assigns to each proposition P a risk value $\langle P \rangle$ with the following properties. If P and Q are any propositions and P(x) becomes a proposition whenever the variable x is replaced by a term denoting an object, then:

(a) $0 \leq \langle P \rangle \leq 1$

(b) $\langle P \wedge Q \rangle = \sup\{\langle P \rangle, \langle Q \rangle\}$

(c) $\langle P \vee Q \rangle = \inf\{\langle P \rangle, \langle Q \rangle\}$

(d) $\langle P \to Q \rangle = \sup\{0, \langle Q \rangle - \langle P \rangle\}$

(e) $\langle \neg P \rangle = 1 - \langle P \rangle$

(f) $\langle \forall x P(x) \rangle = \sup \langle P(a) \rangle$

(g) $\langle \exists x P(x) \rangle = \inf \langle P(a) \rangle$

(h) $\models P$ iff $\langle P \rangle = 0$

(i) any position is acceptable iff its risk value is non-positive.

In (f) and (g) the sup and inf extend over all constant terms a which may be substituted for the variable x, and for (i) any position α in which I am committed by assertions of propositions $P_1, \ldots, P_m$ and you by $Q_1, \ldots, Q_n$ is assigned the risk value $\langle \alpha \rangle = \Sigma \langle P_i \rangle - \Sigma \langle Q_j \rangle$.

With the substitution $1 - \langle P \rangle$ = truth-value of P, these properties yield the truth tables of the infinite-valued Łukasiewicz logic $Ł_\infty$ [16,17] extended to include quantifiers in the natural way.

The risk value of a proposition P may be interpreted as "my" expected loss in asserting P (and my expected gain if you assert P). In particular, if P is a prime proposition $\langle P \rangle$ may be described as the probability (in my opinion) that a trial of the corresponding elementary experiment will give outcome "no", and Theorem 1 may be paraphrased by saying that the speaker (here "I") behaves as though he assigned to each prime proposition a definite subjective probability of outcome "no". This is the reason for the term "probability-definite". Henceforth all speakers will be assumed to be probability-definite.

Let P be a proposition. If, for some speaker S, $\models P$ or $\models \neg P$, i.e. if $\langle P \rangle = 0$ or $\langle P \rangle = 1$, we shall say that the proposition P is dispersion-free or crisp (for S). Like many semantic notions in this language, crispness is a subjective matter - everyone is entitled to his own opinion. However, if in fact all speakers under consideration are agreed we shall say "P is crisp" without qualification. If every prime proposition is crisp for a certain speaker S then it follows easily by recursion that every proposition is crisp for S and properties (b)-(g) in Theorem 1 reduce to the usual truth tables of classical logic. Thus, in so far as the propositions he is prepared to assert are concerned, S behaves as though he were using classical logic. In this way classical logic falls out as a special case.

3. Logical Identities

A sentence P which, in virtue of its form, is true for every speaker is called a logical identity and we write $\vdash P$. In view of Theorem 1 a sentence can be shown to be a logical identity by computing its risk value for an arbitrary speaker. This method is straightforward but often tedious. There is an alternative procedure, independent of Theorem 1, that is based directly on the dialogue interpretation. Consider, for example, a sentence of the form $P \to (Q \to P)$. Suppose I, the proponent P, assert this sentence. You, the opponent O, may admit it, in which case I have "won", or you may, in the first move of the debate, challenge by asserting P whereupon I must reply with $Q \to P$, thus discharging my original commitment. In move 2 you may admit my new assertion, in which case I win, or challenge it by asserting Q when I reply with P. We have now both asserted P. Since my expected loss due to my assertion is balanced by my expected gain due to your assertion we may in move 3 cancel these assertions. This leaves me with no outstanding commitments and hence

certainly in a winning position. It follows that $P\to(Q\to P)$ is a logical identity. The course of debate may be represented as follows:

0 *P* 1	$P\to(Q\to P)$
1 *O* 3	P
1 *P* 2	$Q\to P$
2 *O*	Q
2 *P* 3	P

Each proposition is labelled with a symbol denoting the speaker who asserted it. On the left of this symbol is placed the number of the move at which the assertion was made, and on its right the number of the move at which the incurred commitment was discharged. Even in this simple form this dialogue method suffices to establish a large number of logical identities. With a slight embellishment (roughly, "my" right at any time to name any proposition and require that both speakers assert it) it leads to a method by which at least every logical identity not containing quantifiers may be proved [5].

Various logical identities which we shall need are listed in the following theorem. The derived connective $\leftrightarrow$ used here is defined by the rule: $P\leftrightarrow Q$ is an abbreviation for $(P\to Q)\wedge(Q\to P)$. Whenever $\vdash P\leftrightarrow Q$, the propositions P and Q have equal risk values for every speaker. Such propositions are <u>logically equivalent</u>: i.e. one may be replaced by the other in any sentence without affecting its truth.

<u>Theorem 2</u>. Let P,Q,R be any propositions and let $P(a)$ be a proposition for every object a. Then:

(a) $\vdash (P\vee P)\leftrightarrow P$

(b) $\vdash (P\vee Q)\leftrightarrow (Q\vee P)$

(c) $\vdash [(P\vee Q)\vee R]\leftrightarrow [P\vee(Q\vee R)]$

(d) $\vdash [P\vee(Q\wedge R)]\leftrightarrow [(P\vee Q)\wedge(P\vee R)]$

(e) $\vdash \neg(P\vee Q)\leftrightarrow(\neg P\wedge\neg Q)$

(f-j) the same, exchanging $\wedge$ and $\vee$;

(k) $\vdash P\leftrightarrow\neg\neg P$

(l) $\vdash [(P\vee Q)\leftrightarrow Q]\leftrightarrow(P\to Q)$

(m) $\vdash [(P\to R)\wedge(Q\to R)]\leftrightarrow[(P\vee Q)\to R]$

(n) $\vdash [(R\to P)\wedge(R\to Q)]\leftrightarrow[R\to(P\wedge Q)]$

(o) $\vdash \neg\forall x P(x)\leftrightarrow\exists x\neg P(x)$

(p) $\vdash \neg\exists x P(x)\leftrightarrow\forall x\neg P(x)$

(q) $\vdash \forall x P(x)\to P(a)$

(r) $\vdash P(a)\to\exists x P(x)$

As an illustration of the dialogue method I give the tableau for the proof of (p). (The tableau splits at move 1 corresponding to the two options for *O*. At "moves" 7 and 8 corresponding assertions of *O* and *P* are cancelled.)

0 *P* 1 $\neg\exists x P(x)\leftrightarrow\forall x\neg P(x)$

1 *P* 2	$\neg\exists x P(x)\to\forall x\neg P(x)$
2 *O* 5	$\neg\exists x P(x)$
2 *P* 3	$\forall x\neg P(x)$
3 *P* 4	$\neg P(a)$
4 *O* 8	$P(a)$
4 *P* 7	F
5 *P* 6	$\exists x P(x)$
5 *O* 7	F
6 *P* 8	$P(a)$

1 *P* 2	$\forall x\neg P(x)\to\neg\exists x P(x)$
2 *O* 5	$\forall x\neg P(x)$
2 *P* 3	$\neg\exists x P(x)$
3 *O* 4	$\exists x P(x)$
3 *P* 8	F
4 *O* 7	$P(a)$
5 *O* 6	$\neg P(a)$
6 *P* 7	$P(a)$
6 *O* 8	F

4. Some Derived Connectives

We have already noted that, in $Ł_\infty$, $Q \vee \neg P$ is a stronger assertion than $P \to Q$. Let us write $P \underset{\sim}{\to} Q$ as an abbreviation for $Q \vee \neg P$ and call $\underset{\sim}{\to}$ bold implication. Both $\to$ and $\underset{\sim}{\to}$ reduce for dispersion-free propositions to ordinary classical implication. The latter has thus two natural generalizations in Łuksiewicz logic. The same applies to conjunction and disjunction. We write $P \underset{\sim}{\vee} Q$ as an abbreviation for $(\neg P) \to Q$ and call $\underset{\sim}{\vee}$ bold disjunction. Like $\vee$, $\underset{\sim}{\vee}$ is commutative and associative; but it is not idempotent: i.e. $P \underset{\sim}{\vee} P$ is not equivalent to P. One finds $\langle P \underset{\sim}{\vee} Q \rangle = \sup\{0, \langle P \rangle + \langle Q \rangle - 1\}$, which suggests an alternative and more direct dialogue interpretation: he who asserts $P \underset{\sim}{\vee} Q$ offers to assert both P and Q if his opponent will pay \$1 (or assert F). We write $P \underset{\sim}{\wedge} Q$ as an abbreviation for $\neg(\neg P \underset{\sim}{\vee} \neg Q)$ and call $\underset{\sim}{\wedge}$ bold conjunction. Like $\wedge$, $\underset{\sim}{\wedge}$ is commutative and associative but not idempotent: $P \underset{\sim}{\wedge} P$ is not equivalent to P. One finds $\langle P \underset{\sim}{\wedge} Q \rangle = \inf\{1, \langle P \rangle + \langle Q \rangle\}$, giving the direct interpretation: he who asserts $P \underset{\sim}{\wedge} Q$ agrees either to pay \$1 (or assert F) or to assert both P and Q (at his own choice).

A consideration of truth functions shows that no further equally simple generalizations exist: the classical connectives $\to$, $\wedge$, and $\vee$ have each exactly two simple generalizations in $Ł_\infty$; namely, $\to$ and $\underset{\sim}{\to}$, $\wedge$ and $\underset{\sim}{\wedge}$, and $\vee$ and $\underset{\sim}{\vee}$, respectively.

A vast number of logical identities involving these connectives can easily be obtained. Some which we shall need are given in the following theorem.

Theorem 3. For any propositions P, Q, and R:

(a) $\vdash (P \vee Q) \to (P \underset{\sim}{\vee} Q)$

(b) $\vdash (P \underset{\sim}{\wedge} Q) \to (P \wedge Q)$

(c) $\vdash (P \underset{\sim}{\to} Q) \to (P \to Q)$

(d) $\vdash [P \underset{\sim}{\wedge} (P \to Q)] \to Q$

(e) $\vdash [(P \underset{\sim}{\wedge} Q) \to R] \leftrightarrow [P \to (Q \to R)]$

(f) $\vdash (P \to Q) \leftrightarrow (Q \underset{\sim}{\vee} \neg P)$

(g) $\vdash (P \underset{\sim}{\to} Q) \leftrightarrow (Q \vee \neg P)$

(h) $\vdash \neg(P \underset{\sim}{\vee} Q) \leftrightarrow (\neg P \underset{\sim}{\wedge} \neg Q)$

(i) $\vdash \neg(P \underset{\sim}{\wedge} Q) \leftrightarrow (\neg P \underset{\sim}{\vee} \neg Q)$

(j) $\vdash [(P \to Q) \underset{\sim}{\wedge} (Q \to R)] \to (P \to R)$

(k) $\vdash [(P \leftrightarrow Q) \underset{\sim}{\wedge} (Q \leftrightarrow R)] \to (P \leftrightarrow R)$

(l) $\vdash P \underset{\sim}{\vee} \neg P$ and hence $\vdash \neg(P \underset{\sim}{\wedge} \neg P)$

(m) $\vdash \neg[(P \underset{\sim}{\wedge} \neg Q) \underset{\sim}{\wedge} Q]$

(n) $\vdash [(P \underset{\sim}{\wedge} \neg Q) \underset{\sim}{\vee} (P \wedge Q)] \leftrightarrow P$

(o) $\vdash [(P \underset{\sim}{\wedge} \neg Q) \underset{\sim}{\vee} Q] \leftrightarrow (P \vee Q)$

(p) $\vdash [(P \underset{\sim}{\wedge} Q) \wedge (P \underset{\sim}{\wedge} R)] \leftrightarrow [P \underset{\sim}{\wedge} (Q \wedge R)]$

(q) $\vdash [(P \underset{\sim}{\wedge} Q) \vee (P \underset{\sim}{\wedge} R)] \leftrightarrow [P \underset{\sim}{\wedge} (Q \vee R)]$

(a), (b), and (c) show that $\vee$, $\underset{\sim}{\wedge}$, and $\underset{\sim}{\to}$ are "stronger" connectives than $\underset{\sim}{\vee}$, $\wedge$, and $\to$ respectively; (d) and (e) give two simple uses of $\underset{\sim}{\wedge}$; (f) and (g) show the two forms taken in $Ł_\infty$ by a familiar classical equivalence; (l) is the "principle of the excluded middle", which holds for $\underset{\sim}{\vee}$ although it fails for $\vee$. It can be shown not only that $\underset{\sim}{\wedge}$ and $\underset{\sim}{\vee}$ are associative and commutative but also that (like $\wedge$ and $\vee$) they are both distributive with respect to $\wedge$ and $\vee$; this last property is illustrated by (p) and (q).

5. Fuzzy Sets

It has become customary in mathematics to adopt the notion of a set as a primitive concept. However, much may be said (see, for example, [18],§2) for the altern-

ative of regarding every set as being determined by a property: indeed, unless it be finite, a set can hardly be given except by naming a property which characterizes its members. Now a property is just a (unary) predicate: i.e. a mapping which assigns to each object a proposition. For instance, the set P of prime numbers is determined by the predicate "prime": i.e. by the map which assigns to each number a the proposition " a is prime" - in fact we use the expression " $a \in P$ " simply as an abbreviation for this proposition. Thus to specify any set P amounts to specifying the meaning of the proposition $a \in P$ for every object a . With this approach our new definition of the meaning of a proposition (Definition 2) automatically gives rise to a new definition of "set". In fact, it turns out that in the general case in which we admit dispersive propositions we arrive exactly at Zadeh's notion of a "fuzzy set" [19], while if we allow only crisp propositions we obtain as before the classical notion of a set.

From the present viewpoint, then, the concept of a fuzzy set is prior to that of a set: a set is simply a special kind of fuzzy set. Certainly, in spite of the terminology, a fuzzy set is not a particular kind of set. To emphasize this situation, and to bring out the analogy between fuzzy set theory with Łukasiewicz logic and ordinary set theory with classical logic I shall use the term "sett" as an abbreviation for "fuzzy set". A set now becomes a crisp "sett". We thus arrive at the following definition:

Definition 3. An expression A is said to denote a "sett" whenever a rule is laid down which assigns a proposition, denoted $a \in A$, to every object a . If (for some speaker S) this proposition is crisp for every object a then the "sett" is said to be crisp (for S). A set is a crisp "sett". A "sett" which is not crisp is fuzzy. By "abuse of language" a fuzzy "sett" is also referred to as a fuzzy set.

If an expression P(a) denotes a proposition for every object a then the "sett" $\{x:P(x)\}$ (in which x is any variable) is defined by the stipulation that, for any a , the expression $a \in \{x:P(x)\}$ denotes P(a) . $\{x:F\}$ is the empty "sett", denoted $\emptyset$; $\{x:T\}$ is the universal "sett", denoted I . (T is an abbreviation for $\neg F$, the universally true proposition.) Both $\emptyset$ and I are crisp. $a \notin A$ will be used as an abbreviation for $\neg(a \in A)$.

For practical purposes it is natural to assume that "setts" are distinguishable (for instance syntactically) from objects: i.e., technically, they belong to a different "type". We shall assume for simplicity that objects are all of the same type and call them points. With this understanding "$\in$" denotes a binary predicate or relation between points and "setts". But (cf. the remarks at the beginning of this section) it should not be regarded as a set of ordered pairs - rather, it is a map that assigns to any pair (a,A) the proposition $a \in A$. Moreover, in as much as this proposition is in general dispersive, $\in$ is a fuzzy relation. We shall meet other fuzzy relations later.

Previous discussions of fuzzy sets have relied on a generalization of the notion of the characteristic function of a set. This concept plays no part here, either in the definitions or in the statements and proofs of theorems. However, it is easily defined. Let A be a "sett" and let S be a speaker. Then for each object a the proposition $a \in A$ has, for S , a definite risk value $\langle a \in A \rangle$. The map which assigns to each a the number $\chi_A(a) = 1 - \langle a \in A \rangle$ is called the characteristic function of A for S . (Naturally, the function χ_A is here dependent on the speaker S .)

6. Elementary Operations

Our definition of a "sett" is formally identical to a valid classical definition of a set. The same applies to the following definitions of some standard operations and relations involving "setts".

Definition 4. The intersection $A\cap B$ and union $A\cup B$ of two "setts" A and B are defined to be respectively $\{x:x\in A\wedge x\in B\}$ and $\{x:x\in A\vee x\in B\}$. The complement $\sim A$ of A is $\{x:x\notin A\}$. Further, we define the relations $\subset$ and $=$ between "setts" by:

$A\subset B$ is an abbreviation for $\forall x(x\in A\rightarrow x\in B)$, and

$A=B$ is an abbreviation for $\forall x(x\in A\leftrightarrow x\in B)$.

$A\not\subset B$ and $A\neq B$ will be used as abbreviations for $\neg A\subset B$ and $\neg A=B$.

Notice that, like $\in$, $\subset$ and $=$ are fuzzy relations. The definition of $\subset$, for instance, does not merely describe the circumstances under which $A\subset B$ is true (the necessary and sufficient condition for this is easily seen to be $\chi_A\leq\chi_B$, in agreement with Zadeh [19]), it also associates a definite commitment (and hence a risk value) with the assertion of $A\subset B$ in cases when it is not true. The same applies to $A=B$. It is possible to introduce crisp relations, $\subset_c$ and $=_c$ say, corresponding to $\subset$ and $=$ by writing $A\subset_c B$ for $\models A\subset B$ and $A=_c B$ for $\models A=B$. However, since (in our present formulation) "$\models$" belongs to the metalanguage, so do "$\subset_c$" and "$=_c$". Clearly, any fuzzy predicate can be "crispened" in the same way.

Theorem 4. Let A,B,C be "setts" and let a be a point. Then:

(a) $\vdash A\cup A = A$,

(b) $\vdash A\cup B = B\cup A$,

(c) $\vdash A\cup(B\cup C) = (A\cup B)\cup C$,

(d) $\vdash A\cup(B\cap C) = (A\cup B)\cap(A\cup C)$,

(e) $\vdash \sim(A\cup B) = \sim A\cap\sim B$,

(f-j) the same exchanging $\cap$ and $\cup$,

(k) $\vdash \sim\sim A = A$,

(l) $\vdash A\cup B = B \leftrightarrow A\subset B$,

(m) $\vdash (A\subset C\wedge B\subset C) \leftrightarrow A\cup B\subset C$,

(n) $\vdash (C\subset A\wedge C\subset B) \leftrightarrow C\subset A\cap B$,

(o) $\vdash A\subset A$,

(p) $\vdash (A\subset B\mathbin{\tilde{\wedge}} B\subset C)\rightarrow A\subset C$,

(q) $\vdash A=A$,

(r) $\vdash A=B \leftrightarrow B=A$,

(s) $\vdash (A=B\mathbin{\tilde{\wedge}} B=C)\rightarrow A=C$,

(t) $\vdash A=B\rightarrow(a\in A\leftrightarrow a\in B)$,

(u) $\vdash (a\in A\mathbin{\tilde{\wedge}} A\subset B)\rightarrow a\in B$,

(v) $\vdash A\neq\emptyset \leftrightarrow \exists x(x\in A)$.

(a)-(n) follow easily from the corresponding assertions in Theorem 2. The proofs of (p), (s), and (u) use in addition Theorem 3(j), (k), and (d), respectively. (t) follows immediately from the definition of equality, using Theorem 2(q). The other results are proved with no more difficulty.

At first sight the relations in Theorem 4 seem very familiar, but one must remember that the underlying logic is $\mathfrak{L}_\infty$ and not classical logic. (m), for instance, not only asserts that $A\cup B\subset_c C$ iff both $A\subset_c C$ and $B\subset_c C$ (the classical characterization of the least upper bound); it also says that the risk in asserting $A\cup B\subset C$ is

always equal to the greater of the risks in asserting $A \subset C$ and $B \subset C$. Similarly, (p) does not merely say that if $A \subset_c B$ and $B \subset_c C$ then $A \subset_c C$ (which, along with (o), characterizes a classical partial ordering); it also says that the risk in asserting $A \subset C$ never exceeds the sum of the risks in asserting $A \subset B$ and $B \subset C$.

It is easy to deduce from (t) that if $\vdash A=B$ then A and B are <u>equivalent</u> in that they are interchangeable in any proposition; (a)-(j) provide a number of examples of such equivalent "setts". These identities show that the family of all "setts" is a distributive lattice in the classical sense, the corresponding partial ordering being (in view of (l)) the (crisp) relation which holds between two "setts" A and B iff $\vdash A \subset B$. However, in spite of (e), (j), and (k), $\sim$ is not a classical complementation, since neither $\vdash A \cup \sim A = I$ nor $\vdash A \cap \sim A = \emptyset$ holds in general. What is to be done about this we shall see in the next section.

7. Bold Intersection and Union

In ordinary set theory two sets A and B are said to be disjoint if $A \cap B = \emptyset$. As Zadeh ([19], p. 35) has observed, this relation is much too strong for practical use in fuzzy set theory: for instance, even A and $\sim A$ are not disjoint in this sense. However, let us consider the classically equivalent condition for disjointness, $A \subset \sim B$. By Definition 4, $A \subset \sim B$ is equivalent to $\forall x(x \in A \rightarrow x \notin B)$ which is, by Theorem 3(f), equivalent to $\forall x(x \notin B \vee x \notin A)$ and finally, by Theorem 2(p) and Theorem 3(h), to $\neg \exists x(x \in A \wedge x \in B)$. But, by Theorem 2(k) and Theorem 4(v), this means $\{x: x \in A \wedge x \in B\} = \emptyset$. We shall call the "sett" mentioned here, which in view of Theorem 3(b) is contained in $A \cap B$, the <u>bold intersection</u> $A \underset{\sim}{\cap} B$ of A and B. It and the corresponding bold union provide the key to the concepts of disjointness and difference of "setts":

<u>Definition 5</u>. Let A and B be "setts". The <u>bold intersection</u> $A \underset{\sim}{\cap} B$ and <u>bold union</u> $A \underset{\sim}{\cup} B$ are respectively the "setts" $\{x: x \in A \underset{\sim}{\wedge} x \in B\}$ and $\{x: x \in A \underset{\sim}{\vee} x \in B\}$. The sentence "A <u>and</u> B <u>are (weakly) disjoint</u>", denoted $A \perp B$, is an abbreviation for $A \underset{\sim}{\cap} B = \emptyset$. The <u>difference</u> $A \backslash B$ of A and B is $A \underset{\sim}{\cap} \sim B$.

Of the many logical identities involving these operations, I offer a few that follow easily from Theorem 3, (l)-(q). In the case of crisp "setts", when the distinction between the two kinds of union and intersection can be ignored, they reduce to familiar classical properties of the complement and difference operation on sets.

<u>Theorem 5</u>. Let A, B, C be any setts. Then:

(a) $\vdash A \perp \sim A$: i.e. $A \underset{\sim}{\cap} \sim A = \emptyset$,

(b) $\vdash A \underset{\sim}{\cup} \sim A = I$,

(c) $\vdash A \backslash B \perp B$,

(d) $\vdash (A \backslash B) \underset{\sim}{\cup} (A \cap B) = A$,

(e) $\vdash (A \backslash B) \underset{\sim}{\cup} B = A \cup B$,

(f) $\vdash (A \backslash B) \cap (A \backslash C) = A \backslash (B \cup C)$,

(g) $\vdash (A \backslash B) \cup (A \backslash C) = A \backslash (B \cap C)$.

8. Convex Fuzzy Sets

As a final illustration of the formulation of fuzzy set theory using Łukasiewicz logic I shall now develop the elements of the theory of convex fuzzy sets.[4] To

[4]The notion of a convex fuzzy set was introduced by Zadeh in his original paper [19]. It has been discussed also by Brown [1].

this end we assume that the points are the elements of a linear space.[5] Thus there are now two "types" of object, points and (real) numbers. Particular numbers and variables for numbers will be denoted by Greek letters. Quantification extends only over the indicated type - e.g. $\forall\lambda$ means "for every number λ ", while $\forall x$ means "for every point x ". Our definition of " A is convex" (abbreviated " A conv") is syntactically identical to the classical definition:

Definition 6. Let A be a "sett" in a real linear space. Then A conv is an abbreviation for

$$\forall\lambda\{\lambda\in[0,1] \rightarrow \forall x\forall y[(x\in A \wedge y\in A) \rightarrow \lambda x+(1-\lambda)y\in A]\} \quad .$$

(Here $[0,1]$ denotes the closed unit interval, a crisp "sett" of numbers.) It is easy to show that the condition for the truth of " A conv" agrees with that adopted by Zadeh [19] as the definition of a convex fuzzy set. However, Definition 6 is richer than this: it assigns also a risk value to the assertion A conv in the case when $\not\models$ A conv . Thus one can now speak of "degrees of convexity": for example if $\langle$A conv$\rangle = 0.2$, as for instance when A is the "sett", in a 1-dimensional space, whose characteristic function is sketched in fig. 1 (cf. fig. 4 in Zadeh [19]),

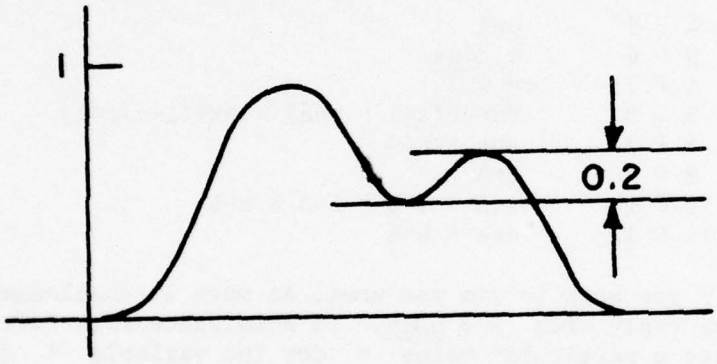


Fig. 1. Characteristic function of a nearly convex sett.

then we might say " A is nearly convex", or more precisely "the truth value of ' A is convex' is 0.8 ".

Zadeh shows that if A and B are convex fuzzy sets then so is $A\cap B$: i.e., in the above notation, if $\models$ A conv and $\models$ B conv then $\models A\cap B$ conv . This result is contained in:

---

[5]Formally this means that the underlying language is enriched by various symbols: for instance binary function symbols for addition and multiplication. The interpretation of these and other nonlogical symbols is not to be understood in terms of orthodox mathematical model theory (which begs the question of interpretation in any practical sense); instead, we assume that to each symbol there corresponds an experimental procedure in such a way that any prime proposition denotes an elementary experiment (Definition 1). (For details see [4,5].) For simplicity we assume that any proposition expressing an arithmetic relation between numbers or a linear relation between points is crisp. (By relaxing these requirements one can obtain a theory incorporating "fuzzy points" and even "fuzzy numbers".) We assume also the usual axioms for a linear space: for example $\vdash \forall x\forall y(x+y=y+x)$.

Theorem 6. For any "setts" A and B

$$\vdash (A \text{ conv} \wedge B \text{ conv}) \rightarrow A \cap B \text{ conv} \quad .$$

But this theorem says more than that: it tells us also that the risk in asserting $A \cap B$ conv never exceeds the greater of the risks in asserting A conv and B conv.

In dialgoue form the proof of the theorem is as follows:

0	*P*	1	$(A \text{ conv} \wedge B \text{ conv}) \rightarrow A \cap B \text{ conv}$
1	*O*	5	$A \text{ conv} \wedge B \text{ conv}$
1	*P*	2	$A \cap B \text{ conv}$
2	*O*	7	$\alpha \in [0,1]$
2	*P*	3	$\forall x \forall y[(x \in A \cap B \wedge y \in A \cap B) \rightarrow \alpha x + (1-\alpha)y \in A \cap B]$
3	*O*	10	$a \in A \cap B \wedge b \in A \cap B$
3	*P*	4	$c \in A \cap B$

Here c is an abbreviation for $\alpha a + (1-\alpha)b$. At this point the tableau splits. We follow one branch:

4	*P*	9	$c \in A$
5	*O*	6	$A \text{ conv}$
6	*P*	7	$\alpha \in [0,1]$
6	*O*	8	$\forall x \forall y[(x \in A \wedge y \in A) \rightarrow \alpha x + (1-\alpha)y \in A]$
8	*P*	12	$a \in A \wedge b \in A$
8	*O*	9	$c \in A$
10	*O*	11	$a \in A \wedge a \in B \wedge b \in A \wedge b \in B$
11	*O*	12	$a \in A \wedge b \in A$

Commentary. If you hope to win you must, at move 1, challenge my initial assertion and I must then reply with $A \cap B$ conv. In accordance with Definition 6 you must now (move 2) choose a particular value α for the variable λ and will lose unless you challenge. At move 3 you must choose particular points a and b and are again forced to challenge. By Definition 4 $c \in A \cap B$ means $c \in A \wedge c \in B$. At move 4 you must choose which assertion I am to make; without loss of generality we may assume you choose the former. At move 5 I refer to your assertion in move 1 and require you to assert A conv . By Definition 6 I may now (move 6) choose the same number α as you chose at move 2, and challenge. Since my assertion here coincides with yours at move 2 these two assertions may be cancelled (move 7). Next (move 8) I choose the same points a and b as you chose at move 3 and challenge again. My assertion in move 4 is now cancelled (move 9) by yours in move 8. In view of the definition of $\cap$ your assertion at move 3 may be expanded as in move 10 (where we omit parentheses in view of the associative law, Theorem 2(h)); and this, in view of the definition of $\wedge$, may be replaced (at my request) by the assertion in move 11. Finally, in move 12, this assertion cancels my assertion in move 8. I am left with no outstanding commitments, and so have reached a winning position. This shows that my original assertion was safe, which proves the theorem.

REFERENCES

[1] J.G. Brown, "A note on fuzzy sets", Information and Control, vol. 18, pp. 32-39, 1971.

[2] C.C. Chang, "Algebraic analysis of many valued logics", Trans. Amer. Math. Soc.,

vol. 88, pp. 467-490, 1958.

[3] C.C. Chang, "Infinite valued logic as a basis for set theory", Proc. 1964 Int. Congress for Logic, Methodology and Philosophy of Science, Jerusalem, August 1964, North-Holland.

[4] R. Giles, "A nonclassical logic for physics", Studia Logica, vol. 33, part 4, 1974.

[5] R. Giles, "A pragmatic approach to the formalization of empirical theories", to appear in Proc. Conf. Formal. Methods in the Methodology of Empirical Sciences, Warsaw, June 1974.

[6] R. Giles, "A logic for subjective belief", to appear in Proc. Conf. on Foundations of Probability and Statistics and Statistical Theories of Science, London, Ontario, May 1973.

[7] R. Giles, unpublished.

[8] J.A. Goguen, "L-fuzzy sets", J. Math. Anal. Appl. vol. 18, pp. 145-174, 1967.

[9] J.A. Goguen, "The logic of inexact concepts", Synthese, vol. 19, pp. 325-373, 1968-69.

[10] D. Klaua, "Über einen Ansatz zur mehrwertigen Mengenlehre", Monatsb. Deutsch. Akad. Wiss. Berlin, vol. 7, pp. 859-867, 1965.

[11] D. Klaua, "Über einen zweiten Ansatz zur mehrwertigen Mengenlehre", Monatsb. Deutsch. Akad. Wiss. Berlin, vol. 8, pp. 161-177, 1966.

[12] D. Klaua, "Grundbegriffe einer mehrwertigen Mengenlehre", Monatsb. Deutsch. Akad. Wiss. Berlin, vol. 8, pp. 782-802, 1966.

[13] D. Klaua, "Ein Ansatz zur mehrwertigen Mengenlehre", Math. Nachr. vol. 33, pp. 273-296, 1967.

[14] D. Klaua, "Einbettung der klassischen Mengenlehre in die mehrwertige", Monatsb. Deutsch. Akad. Wiss. Berlin, vol. 9, pp. 258-272, 1967.

[15] D. Klaua, "Stetige Gleichmächtigkeiten kontinuierlich-wertiger Mengen", Monatsb. Deutsch. Akad. Wiss. Berlin, vol. 12, pp. 749-758, 1970.

[16] J. Łukasiewicz and A. Tarski, 1930, "Investigations in the sentential calculus", Translation in J. Łukasiewicz, Selected Works, Reidel, Amsterdam, 1970; and in A. Tarski, Logic, Semantics, Metamathematics, Clarendon Press, Oxford, 1956.

[17] N. Rescher, Many-valued Logic, McGraw-Hill, New York, 1969.

[18] H. Weyl, Philosophy of Mathematics and Natural Science, Princeton, 1949.

[19] L.A. Zadeh, "Fuzzy sets", Information and Control, vol. 8, pp. 338-353, 1965.

CONJECTURES ON MANY-VALUED LOGIC, REGIONS, AND CRITERIA FOR CONFLICT RESOLUTION[1]

Stephen Gale
University of Pennsylvania
U.S.A.

1. Though a general theory of the appropriate uses of varieties of formal languages is still in its formative stages, recent developments in non-classical logics are beginning to have important impacts on the structure of scientific description and reasoning. Thus, as Carnap's (10:51) "Principle of Tolerance" has become (implicitly) a serious methodological consideration, scientists are now beginning to question the "linguistic" foundations of their measurement and inference procedures and, in general, to regard the place of alternative formal languages in the modelling process as an open issue. In part the moves in this direction have been adoptive: new languages are available and, for experimental purposes, they are used in a variety of contexts. More importantly, however, several of the changes in orientation have been based on failures by existing formalizations to account for specific classes of questions.

This paper summarizes and extends some recent comments on the use of many-valued logics in treating the problem of territorial regionalization and its concomitant effects on criteria for conflict resolution (16, 17, 19). The motivation stems from a pragmatic concern with efficacy of the kinds of formal languages which have been used to prescribe geographic partitions and the ways in which these partitions, in turn, affect the outcomes of societal decisions. The argument parallels a number of earlier discussions concerning the rationale for using non-classical formal languages[2] and uses the theory of fuzzy sets (56) as one representation of the set-theoretic characterization of the properties of a class of non-classical languages. Following from these earlier discussions, then, the argument will be developed first (§§2-3) in terms of some general remarks on a language-based program for scientific inquiry. The second step (§§4-13) will provide a characterization of the notion of "region" within many-valued logic and demonstrate its association with a fuzzy set-theoretic interpretation. The final step (§§14-17) will extend these descriptive results to some comments on their effect on the design (and legitimacy) of criteria for conflict resolution; in particular, the contention is that the characterization presented will provide grounds for the use of weighted-voting procedures.

2. The term "model" has come to have about as many different meanings as there are people doing modelling.[3] Quite apart from the various functions (e.g., simplification and partitioning of problems), however, at its root a model is simply an expression in a given language: there are linguistic models of thought (e.g., natural languages), mathematical models of entitivity and relations (e.g., various set theories), physical models (e.g., reduced scale representations of streams, airfoils, etc.), and so on. Within the framework of a given language, a model provides a

(usually simplified) structural representation of some fragment of concepts and/or phenomena. Of course, except for special abstract cases (e.g., those arising in connection with classical model theory), the representational rules are rarely complete; they simply form conditions for satisfaction which (under some circumstances) provide the basis for successively better approximations.

More specifically, the modelling process may be conceived of as a question-answering system $\underline{\underline{Z}} = {}_{D}\langle \underline{\underline{Q}}, \underline{\underline{M}}, \underline{\underline{G}} \rangle$ consisting of a question ($\underline{\underline{Q}}$), an answer (or model $\underline{\underline{M}}$), and an algorithm ($\underline{\underline{G}}$) which describes the relations between $\underline{\underline{Q}}$ and $\underline{\underline{M}}$ (18). Formally, we thus have

$$\underline{\underline{Q}} = {}_{D}\langle \underline{L}, \underline{A}, \underline{(?)S}, \underline{P} \rangle$$

and

$$\underline{\underline{M}} = {}_{D}\langle \underline{L}, \underline{A}, \underline{T}, \underline{U}, \underline{R}_1, \ldots, \underline{R}_n \rangle$$

where $\underline{L}$ is a given (formal or natural) language, $\underline{A}$ is a set of axioms (consisting specifically of those postulates relating to conditions for measurement), $\underline{(?)S}$ is an interrogative sentence (or proposition), $\underline{P}$ is a set of presuppositions, $\underline{T}$ is a set of potential answers to (i.e., theories about) $\underline{\underline{Q}}$ (as determined from $\underline{P}$), $\underline{U}$ is the (non-empty) universe of discourse, and $\underline{R}_1, \ldots, \underline{R}_n$ are relations on $\underline{U}$. $\underline{\underline{G}}$ may be regarded as a (stochastic or fuzzy) linear automaton (e.g., 53) which provides a context-dependent algorithm for the recursive elimination and substitution of unsatisfactory elements $t_i \in \underline{T}$. For the present, the designation of rules of satisfaction and partial satisfaction is regarded as a metatheoretical issue, the solution to which is dependent (at least) on the nature of $\underline{\underline{Q}}$ and the properties of $\underline{L}$.

3. Clearly, this very abstract perspective on inquiry processes needs much more flesh than has been given here. For the present purposes, however, one point in the analysis is of special importance: $\underline{\underline{Z}}$ is principally dependent on its specification under a language, $\underline{L}$. Questions, for example, arise and are phrased in a language; the language of the answers (or models) must be conformable (in the sense that a response in some other language is not intelligible without explicit translation rules); and the relationships between $\underline{\underline{Q}}$ and $\underline{\underline{M}}$, i.e., $\underline{\underline{G}}$, must be sufficiently rich to be able to account for (at least) the syntax and semantics of all sentences (propositions) within the inquiry process. Language, then, is the concept with the principal epistemic force. Ideas and the object world itself are treated as <u>linguistically based</u> entities (within a question-answering process); the reconstruction of ideas and the addition of new concepts (e.g., value-based propositions) are similarly regarded as being predicated on the reformulation of the underlying linguistic structure. The analysis employed here thus consists primarily of a re-examination of the nature of the formal languages used in the description of regions.

4. In Western tradition, the concepts of "region" and "boundary" arise in connection with two related developments: (a) the organization of space under conditions where property can be owned, and (b) the view of a legal system which resolves conflicts within territorially legitimatized criteria of justice (17, 19). Regions (whether political or administrative) extend to cover the surface and, at the interface, by virtue of the meaning of contiguity, is the boundary. It is a subtle move, based on a view of the efficacy of a given language and one with enormous explanatory and

normative force.

The tradition here is classical: regions and their boundaries (legal, functional or otherwise) are real and formed from a partition of land areas into mutually separable units which completely cover geographic space. Georgescu-Roegen's (21) term "arithmomorphic" will be adopted as the designation for this conception of "region." It is meant to convey an impression of crisp, clearly identified units within a tradition of legally separable areal entities.

5. A parallel tradition on the notions of territory and boundary can also be identified -- though not with an equivalent historical or legal foundation. In fact, it arises primarily in the study of primitive societies, animal territories, problems of pattern recognition, and the like.[4] Two examples should provide some grounding:

> The regional concept is a static view of human life, in two senses: (a) first, a regional system has validity for the moment at which it is devised and for no other moment ... and (b) second, regional studies have tended to treat the defined region as a community isolated from the rest of the world yet clearly no area or region in the modern world is independent of other parts of the world. (24: 471)

> Territorial patterns... take on a great variety of forms. There are group or horde territories and there are individual territories. There are fixed and portable territories. Some territories are clearly marked,... and are always defended; others are very "fuzzy" and may be defended only under specific circumstances. (45: 23)

Once again using Georgescu-Roegen's (21) terminology, this view will be designated as the "non-arithmomorphic" tradition of territorial conceptions. The term is meant to imply a state of flux, indeterminacy, and context-dependence; it is neither crisp nor does it (necessarily) imply a completely partitioned set of territories.

6. In their non-formal senses, then, "region" and "boundary" convey at least two kinds of presuppositions. On the one hand, it is intended to account for the meaning of classical geometric concepts as they are applied to territorial partitions. And on the other hand, there is a rather less deterministic concept in which regions are inexact and ambiguous and defined only with respect to specific functional relationships. Neither usage is explicitly normative; neither are they entirely distinguishable in natural language. As with most such concepts, meaning is dependent not only on the syntax and semantics, but also on the pragmatic aspects of language.

7. Now, given the two senses of "region" described above (§§4-6), languages must be distinguished for use in the inquiry system $\underline{\underline{Z}}$. For the arithmomorphic case the problem may be regarded as solved: the language of description and inference is a formal first-order functional calculus (with identity); the associated set theory is, of course, the usual Boolean set theory, say with the Zermelo-Fraenkel axiomitization. The sense of the language thus reifies the Aristotelian notions of identity, noncontradiction, and the principle of the excluded middle.

For the non-arithmomorphic case, however, the designation of a suitable formal language is not clear-cut. As Lewis (31: 483) puts it, "there are an unlimited number

of possible systems of logic, each such that every one of its laws is true and is applicable to deducation." In previous discussions (16, 17, 19), use has been made of the concepts of fuzzy sets (56) and tolerance spaces (41) to provide set theoretic and geometric characterizations of "region" and "boundary" (respectively). The argument was, in effect, an attempt to demonstrate that the non-arithmomorphic senses of these terms could be modelled within the structure of a non-classical set theory and geometry. By appealing to a direct translation of the meaning of "region" (qua sense of "region"), a point-by-point assignment of points to a fuzzy set with a generalized characteristic function on the unit interval [0,1] was used to designate the "members" of a region;[5] a similar move was made for the idea of a "boundary" with the points within some ε-tolerance being designated as the boundaries of fuzzy regions. Finally, given the set-theoretic account, the character of the underlying formal language was presented as a class of many-valued alethic calculi.

8. Though I believe that the previous arguments have some practical justification, some of the important language-based insights have nevertheless been side-stepped. Appeal must also be made to some more fundamental presuppositions concerning the kinds of languages used in developing models, say something akin to Łukasiewicz's (32) early motivation for 3-valued calculi as being representative of a non-deterministic philosophy; or something on the order of Watanabe's (51) argument for the use of modular logics for description and inference in quantum mechanics.

Three reasons will be given for considering non-classical languages for describing and reasoning about the concept of "region." In particular, appeal will be made to certain dissatisfactions which arise in the use of languages which imply deterministic knowledge, problems about the categoricity of models for "region," and some analogies to other fields which have proposed the use of alternative languages. The effect will, of course, be to provide an outline of complementary arguments in support of the fuzzy set-theoretic perspective already advanced.

9. The first argument parallels the set-theoretic characterization of "region" in the sense that it appeals to the grounds for describing the term's meaning for indeterminate and inexact classes of conditions.

Except in those cases where for, say, legal or administrative purposes, regions are formed with respect to exact criteria, territories based on interaction patterns and the boundaries between them are often fluid or fuzzy. The designation of a neighborhood, for example, often shifts with respect to the kinds of people that live in the general area; and, more importantly, the designation is also indeterminate with respect to even the kinds of day-to-day patterns of functional interchanges. All these leave the neighborhood as a concept with a non-specific class of members -- and, equally significantly, with a (temporally-dependent) shifting class of issues on which its designation is based. In effect, it is a fuzzy concept, a non-arithmomorphic version of "region" which uses a non-classical descriptive (and therefore inferential) language as its basis.

One argument concerning the character of the formal language underlying this view of region can be drawn by reference to Łukasiewicz's (32) discussion on the nature of determinism. For any individual, $x \in \underline{X}$, in a society (such as a city) fuzzy or non-fuzzy "belonging" to a neighborhood, $\underline{N}$, can be evaluated only with respect to a given context and for a sequence of behaviors at past times, t-1, t-2,... . Barring

a completely deterministic universe, however, for future times, belonging is in principle indeterminate since neither can an individual's interaction pattern be causally explicated nor can the relevant aspects of the society (which combine to give designation to a neighborhood) be completely known. Thus, for the past, any individual $x \in \underline{X}$ can be said either to be a member or not a member of $\underline{N}$ (i.e., $x \in \underline{N}_{t-1}$ or $x \notin \underline{N}_{t-i}$ $(i = 1, 2, \ldots)$), and depending on extensional evaluation, the truth value T is assigned to the former case and F to the latter. Of the future, however, in general the membership condition is not known (i.e., neither $x \in \underline{N}_{t+i}$ nor $x \notin \underline{N}_{t+i}$ can be assigned a value of T or F); in this case membership is non-classical and is therefore assigned a third value, say I (for indeterminate).[6]

Note that the case of $\underline{N}_{t_1}$ for $i = 0$ has not been treated. In part this is because it presents the kind of tricky metaphysical problems that only philosophers find extremely pertinent,[7] though it is clearly relevant to the fuzzy set-theoretic characterization. More to the point, however, for our purposes it is unnecessary since, by virtue of the argument just given, a sufficient case for the extension of a two-valued to a many-valued language which covers the non-arithmomorphic case of membership has already been introduced.

For the present, the question of precisely what form the language of regionalization will take is left open. However, it should be kept in mind (i) that for non-deterministic contexts at least a 3-valued logic is required; (ii) that the language of regionalization should include 2-valued language as a special case; and (iii) that the change in the language must be interpretable in terms of a change in the conditions for membership. (In fact, this final point will be used as the basis for designating the particular type of many-valued logic to be used.)

10. The second argument arises in connection with the non-categoricity of models of finite numbers of regions under a language of geometry which includes the set membership relation " $\in$" (49).

Consider the usual implications of membership in a region. Given a partition of some space $\underline{K}$ into mutually exclusive and exhaustive members, $K_i \in \underline{K}$ $(i = 1, \ldots, m)$, membership is defined for each individual place $(x,y) \in \underline{\underline{X}}$ as $(x,y) \in K_i$ or $(x,y) \notin K_i$. An individual place either belongs to a region K_i or it does not and it must belong to one and only one.

Even within this perspective of regionalization, membership in a region is viewed not just as a process of assignment, but as a combined process of partition <u>plus</u> assignment. And, though the language of assignment may be arithmomorphic, the formal languages used in modelling the partitions and the partitioning process need not be. Thus, as opposed to the assignment problem for membership in a finite number of subsets (and here we are still affected by the discussion in § 9), with an infinite number of models of regions (i.e., the subsets K_i of $\underline{K}$) there is in general no unique partitioning.

11. The final argument follows from two observations from rather dissimilar sources dealing with the compatability of alternative formal languages.[8] In the interests of brevity, the relevant sources will simply be noted.

First, at least one school of thought on the structure of physical models holds (for a variety of reasons) that there are two separate languages, one for the microcosmic world and one for the macrocosmic (e.g., 9; 37; 35; 25; 8; 26). Second, there are the case studies which report dissimilar (but complementary) bases for information processing behavior in human patients in which surgical disconnection of the cerebral hemispheres has been effected (e.g., 20; 47). Though grounded on quite different theories and observations, remarkably these arguments speak to much the same point: whatever the a priori validity of one or another formal language, for certain classes of problems there seem to be different (and complementary) languages of use. Or, in effect, there are certain classes of statements (in both the physical and psychic domains, for example) which cannot be made within one or another formal language. Note that this claim neither discounts the effectiveness of classical logic as a metalanguage for truth-functional reasoning nor rejects the possibility of translation rules between languages.

12. In each of these arguments, there is an explicit criticism of a single-minded orthodoxy in the use of classes of models and languages and an implicit suggestion that a form of heterodoxy in our use of language may be valuable. These arguments may be summarized as three important claims about the use of formal languages in the analysis of questions concerning regionalization.

A. It should include a 3- (possibly n-) valued alethic language. (This follows from both the set-theoretic characterization in terms of fuzzy sets and the Łukasiewicz-type argument concerning the 3-valuedness of languages for non-deterministic worlds.)

B. The usual 2-valued language should be embedded in the new language. (This follows from the requirements of some kinds of assignment processes and the need for a compatible, alethic metalanguage.)

C. That there is no need to appeal to a single language of description and inference. (This follows from the arguments concerning the multiplicity of languages for different classes of problems.)

These claims are effectively design criteria and their implications go a long way toward specifying the structural properties of the desired language.

13. Though many of the arguments concerning the use of many-valued logics often stress primarily the ontological relativity of language (61: 124), such arguments provide no direct grounds for selecting among the multiplicity of many-valued logics (or for that matter, the construction of new logics). But rather than fight out the issue on ontological grounds,[9] it appears more efficacious to regard the operational structure of "region" in terms of fuzzy sets as the principal criterion for such a selection. Thus, if the codomain of the generalized characteristic function of a fuzzy set is $[0,1]$, then a language of the form of (what Rescher (40: 37-40) terms) Łukasiewicz's $Ł_{\aleph_1}$ can be used as a basis.[10] The truth values of the propositions of this language are the real numbers on the interval $[0,1]$; the assignment rules are given as (40: 336-337):

(1) $/\neg p/ = 1 - /p/$

(2) $/p \wedge q/ = \min(/p/, /q/)$

(3) $/p \vee q/ = \max\ (/p/,\ /q/)$

(4) $/p \rightarrow q/ = \begin{cases} 1 \\ 1 - /p/ + /q/ \end{cases}$ according as $\begin{cases} /p/ \leq /q/ \\ /p/ > /q/ \end{cases}$

(5) $/p \leftrightarrow q/ = /(p \rightarrow q) \wedge (q \rightarrow p)/.$

The symbol /•/ designates the value of the proposition. The same rules also apply to any denumerable or finite partition of the truth values; for example Łukasiewicz's 3-valued logic Ł3 (40: 22-28) is obtained when the interval [0,1] is partioned into the three subsets

(i) $[\ \alpha, 1\ ] \rightarrow$ T

(ii) $(\ \alpha, \beta\ ) \rightarrow$ I

(iii) $[\ 0, \beta\ ] \rightarrow$ F

where the real numbers $\alpha, \beta \in (0,1)$ are given some context-dependent specification. For expository purposes, the family of languages $Ł_3 - Ł_1$ will be called $\underline{\underline{Ł}}$.

Now at least on ontological grounds, there are clearly weaknesses in this characterization. As is well-known, there are not only other languages having truth values consisting of the real numbers on the interval [0,1], but also a reasonable variety of characterizations of 3-, 4-, ..., and n-valued logics (40). Ontological argument could perhaps serve to distinguish one or another system (or give further grounds for the adoption of the Łukasiewiczian systems). But, as Lewis (31: 483-484) puts it, "the unlimited multiplicity and variety of such systems [logics] and principles transcends the limits of human apprehension: practically it is necessary to make a choice amongst them, if there is to be any canon of inference at all." The practical grounds suggested here are simply that $\underline{\underline{Ł}}$ (i) provide a characterization of the fuzzy set-theoretic characterization of region, (ii) prescribe languages in which classical 2-valued calculi are embedded, and (iii) entail sufficient flexibility for a wide variety of alternative systems. No ultimate justification has been given (if, indeed, one could be); just a sufficient justification to cover the general arguments concerning the nature of the term "region."

14. As a characterization of the descriptive qualities of "region," models formed with respect to $\underline{\underline{Ł}}$ together with axioms for fuzzy sets and tolerance geometry clearly have some very desirable intuitive properties. But as it has thus far been presented, a model is simply a way of representing the structural features of the content of questions and answers, and here only with regard to the implications of its language-based features. For example, in itself the formalization gives no real hints as to how a reconceptualization of "region" has anything to do with the practical consequences of the ways in which regions function in the world (e.g., the ways in which social conflicts arise and are resolved).

Though there are other examples of decision procedures which are affected by regionalization, (e.g., regional resource allocation, the judicial process, etc.), voting is probably the most important. Not only is it regarded as the central criterion of democracy, at least in Western countries its almost <u>a priori</u> legitimacy makes it difficult to conceive of what "civilized society" would mean in its absence.

In this regard it therefore is strange to find a comment such as Coleman's (11: 1076):

> ...Increasing geographic mobility... creates a fundamentally different infrastructure to society, one which is not based on a geographic unit. Thus geographically based representation is, or will soon be, an anachronism compared to other systems for effecting the same constituent-representative interchange, systems that have not yet been invented.

In the spirit of the transformation from barter to market economies, Coleman does offer one such invention: political money. But this "invention" is in its way a substitute for the legitimizing criteria of voting in that it gives primary credence to the efficacy of the market mechanism. Decision is there, but voting in the sense of individual franchise and participation is not.

Coleman has, however, touched on the rawest nerve of the voting process. As a means for societal decision-making, voting can be dominated by the ways in which names are assigned to districts (e.g., the outcome of any voting scheme is affected by what we call "gerrymandering"). Realizing that these inequities come hand-in-hand with a regionally-based participation process, Coleman turned to analogies to the market for a solution. And, as might be expected, the transformation led to another way of justifying a form of weighted voting based on an exchange medium for votes.

15. A recent paper on the "regionalization problem" (17) has argued for much the same conclusions as Coleman, though for somewhat different reasons. By recognizing the non-categoricity of models of regional partitions under arithmomorphic models, it was proposed that participation in the political system could be best effected by institutionalizing weighted voting based on the value of the characteristic function of the fuzzy set description of a region. That is, by defining a region in terms of a non-arithmomorphic model, it was suggested that the set of measures for each individual with respect to each (fuzzily defined) region could be interpreted as indices of weighted participation. Votes could then be distributed in accordance with these measures (up to one full "participation measure"), thereby accounting for the variation in regionally-oriented interaction patterns, and decisions were determined in much the same manner as threshold functions (with a tolerance) are employed in connection with switching systems (27: 28-36). Legitimacy, however, was not a function of equity considerations with respect to distributional criteria; following Rawls' (36) argument, it was based only on the processual consideration of <u>fairness</u>.

16. Taken within the context of the arguments concerning the language of regionalization, the meaning of these claims about alternative voting procedures can now be given more substance.

First recall the force of the transfer from the use of an arithmomorphic to a non-arithmomorphic language of "region." "Belonging" or "participating" are, for example, transformed into sets of measures indicating individuals' assessments; areal partitions for voting are, on the other hand, defined relative to the kind of partition inherent in the order of the valuation rules of the language (i.e., 2-, 3-, ... valued). Thus, apart from legal or administrative jurisdictions, characteristics of "belonging" are not regarded as derived solely from locational attributes; "belonging" is an outcome of individual membership conditions, expressed in a given language, and

subject to revision and reassessment on these grounds.

Following from earlier arguments on this issue (17, 19) we may view the impact of this language-based change on voting procedures as shifting our attention from issues concerning the effects of sovereignty, political equality, and the like, on the outcomes of elections[11] (since these are now by-products of the language), to questions about the nature of participation in specific contexts. But even more importantly, it also provides a direct rationale for treating intensities of preferences, since the non-arithmomorphic characterization of "region" is just this: a set of measures which indicate, say, our subjective condition (i.e., intensity) of membership. This is a crucial step, particularly insofar as it rejects the notion that people are "bound to the land" in some neo-feudal sense; regions have operational functions (e.g., in terms of police and fire districts, the jurisdiction of courts, and so on), but individual membership in these regions is not characterized by the usual arithmomorphic conditions based on location. The conclusion, then, is that the change from a 2-valued to a many-valued language for regionalization has as an immediate consequence a weighted voting procedure where the justification for this procedure need not appeal to anything but the characteristics of the language in which the regions are described. The language of regionalization thus directly implies what we mean by voting-based forms of conflict resolution.

17. The basis of the preceding arguments can be summarized as follows:

(i) The process of scientific inquiry, when viewed as a question-answering system, provides for the explicit inclusion of questions of all types and, therefore, of models based on both formal and non-formal languages -- e.g., $\underline{L}$ can include 2-valued and many-valued calculi.

(ii) The notion of "region" can be regarded as having two different natural language meanings: one based on clear-cut partitions of space and the other based on classes of syntactic, semantic, and pragmatic indeterminacies. The former is termed "arithmomorphic", the latter "non-arithmomorphic."

(iii) The arithmomorphic notion of a boundary was identified in terms of a model in which $\underline{L}$ is a two-valued, classical logic and $\underline{A}$ is a set of axioms containing, say, the Zermelo-Fraenkel axioms for set theory and the axioms of Euclidean geometry; the non-arithmomorphic notion of a boundary was identified in terms of a model in which $\underline{L}$ is a many-valued logical calculus based on Lukasiewicz's language $\mathfrak{L}_{\aleph_1}$ (or some finite partition of it) and $\underline{A}$ contains axioms for fuzzy sets and a tolerance geometry.

(iv) Criteria for conflict resolution were described as being based on the properties of the language for describing regions and, using voting as an illustration, it was demonstrated that a many-valued language for regionalization implies the use of weighted voting procedures.

NOTES

1. The partial support of the National Science Foundation, Grant No. GS-39837, is gratefully acknowledged.

2. See Rescher (40) for a detailed discussion of the historical development of many-valued logic.

3. See Suppes (48) for a discussion of some of this variety; note, however, that in Suppes' view all the uses are regarded as effectively equivalent to the model-theoretic conception developed with respect to the semantics of mathematical logic.

4. See, for example, Bohannon (7), Ardrey (1), Sommer (46), and Watanabe (52).

5. This argument has, of course, been dealt with in some detail in the growing literature on fuzzy sets. See, for example, Zadeh (56, 57, 58, 59, 60), Goguen (22), Wee and Fu (53), Bellman and Zadeh (5), Santos (42, 43, 44), deLuca and Termini (13, 14), Bellman (3), Gale (16), Preparata and Yeh (34), Gottinger (23).

6. Note that this move effectively side-steps the Montague-type (33) intensional semantics for tense by simply treating the extensional valuation in a non-classical way.

7. See, for example, Black (6), Khatchadorian (29), Körner (30), Verma (50).

8. Note that I have refrained from appealing to differences in the structure of descriptive and normative statements (i.e., Hume's (28) so-called "separation thesis"). Though I believe this point is well taken in the present context, it would nevertheless bring out a host of philosophical problems which need not concern us at this point.

9. For other discussions of this issue see, for example, Weiss (54, 55), Lewis (31), Reiser (38, 39), Putnam (35), Zinov'ev (61), Rescher (40).

10. Note that, as do Bellman and Giertz (4), I have some reservations about the description of the characteristics of the valuation of the negation operator for fuzzy sets; I agree, however that (p. 156) "the rule...appears quite reasonable in practical applications."

11. See, for example, Dahl (12), Banzhaff (2), Fishburn (15: Chapter 5).

REFERENCES

(1) R. Ardrey, The Territorial Imperative, Atheneum, 1966.

(2) J.F. Banzhaf III, "Weighted Voting Doesn't Work: A Mathematical Analysis," Rutgers Law Review, vol. 19, pp. 317-343, 1965.

(3) R.E. Bellman, Communication, Ambiguity and Understanding. Technical Report No. 72-7, Department of Electrical Engineering, University of Southern California, Los Angeles.

(4) R.E. Bellman and M. Giertz, "On the Analytic Formalism of the Theory of Fuzzy Sets," Information Sciences, vol. 5, pp. 149-156, 1973.

(5) R.E. Bellman and L.A. Zadeh, "Decision-Making in a Fuzzy Environment," Management Science, vol. 17, pp. B-141-B-164, 1970.

(6) M. Black, "Reasoning with Loose Concept," Dialogue, vol. 2, pp. 1-12, 1963.

(7) P. Bohannan, "Space and Territoriality," in P. Bohannan, Africa and Africans, The Natural History Press, pp. 174-182, 1964.

(8) D. Bohm, "Classical and Non-Classical Concepts in the Quantum Theory," British Journal for the Philosophy of Science," vol. 12, pp. 265-280, 1961-62.

(9) N. Bohr, Atomic Theory and the Description of Nature, Cambridge University Press, 1934.

(10) R. Carnap, The Logical Syntax of Language, Routledge and Kegan Paul, 1937.

(11) J.S. Coleman, "Political Money," American Political Science Review, vol. 64, pp. 1074-1087, 1970.

(12) R.A. Dahl, A Preface to Democratic Theory, University of Chicago Press, 1956.

(13) A. de Luca and S. Termini, "Algorithmic Aspects in Complex Analysis," Scientia, vol. 106, pp. 659-671, 1971.

(14) A. de Luca and S. Termini, "Algebraic Properties of Fuzzy Sets," Journal of Mathematical Analysis and Applications, vol. 40, pp. 373-386, 1972.

(15) P.C. Fishburn, The Theory of Social Choice, Princeton University Press, 1973.

(16) S. Gale, "Inexactness, Fuzzy Sets and the Foundations of Behavioral Geography," Geographical Analysis, Vol. 4, pp. 337-349, 1972.

(17) S. Gale, "A Resolution of the Regionalization Problem and its Implications for Political Geography and Social Justice," Working Paper No. 3, Research on Metropolitan Change and Conflict Resolution, Peace Science Department, University of Pennsylvania, 1974.

(18) S. Gale, "A Prolegomenon to an Interrogative Theory of Scientific Inquiry," Working Paper No. 9, Research on Metropolitan Change and Conflict Resolution, Peace Science Department, University of Pennsylvania, 1974.

(19) S. Gale, "Boundaries, Tolerance Spaces and Criteria for Conflict Resolution," Journal of Peace Science, in press.

(20) M.S. Gazzaniga, J.E. Bogen, and R.W. Sperry, "Observations on Visual Perception after Disconnexion of the Cerebral Hemispheres in Man," Brain, vol. 88 (Part 2), pp. 221-236, 1965.

(21) N. Georgescu-Roegen, The Entropy Law and the Economic Process, Harvard University Press, 1972.

(22) J.A. Goguen, "The Logic of Inexact Concepts," Synthese, vol. 19, pp. 325-373, 1968-1969.

(23) H-W. Göttinger, "Toward Fuzzy Reasoning in the Behavioral Sciences," in W. Leinfellner and E. Köhler (eds.), Developments in the Methodology of Social Science, D. Reidel, pp. 287-308, 1974.

(24) D. Grigg, "Regions, Models and Classes," in R.J. Chorley and P. Haggett (eds.), Models in Geography, Methuen, pp. 461-509, 1967.

(25) N.R. Hanson, "The Copenhagen Interpretation of Quantum Mechanics," American Journal of Physics, vol. 27, pp. 1-15, 1959.

(26) P. Heelan, Quantum Mechanics and Objectivity, Nijhoff, 1965.

(27) S.-T. Hu, Threshold Logic, University of California Press, 1965.

(28) D. Hume, Treatise on Human Nature, Oxford University Press, 1969 ed.

(29) H. Khatchadorian, "Vagueness, Meaning and Absurdity," American Philosophical Quarterly, vol. 2, pp. 119-129, 1965.

(30) S. Körner, Experience and Theory, Humanities Press, 1966.

(31) C.I. Lewis, "Alternative Systems of Logic," The Monist, vol. XLII, pp. 481-507, 1932.

(32) J. Łukasiewicz, "O Determinizmie" (On Determinism), English translation in S. McCall (ed.), Polish Logic: 1920-1939, Oxford University Press, pp. 19-39, 1971.

(33) R. Montague, "Logical Necessity, Physical Necessity, Ethics, and Quantifiers," Inquiry, vol. 3, pp. 259-269, 1960.

(34) F.P. Preparata and R.T. Yeh, "Continuously Valued Logic," Journal of Computer and Systems Sciences, vol. 6, pp. 397-418, 1972.

(35) H. Putnam, "Three-Valued Logic," Philosophical Studies, vol. 8, pp. 73-80, 1957.

(36) J. Rawls, "Justice as Fairness," The Philosophical Review, vol. 57, pp. 164-194, 1958.

(37) H. Reichenbach, Philosophic Foundations of Quantum Mechanics, University of California Press, 1944.

(38) O.L. Reiser, "Non-Aristotelian Logics," The Monist, vol. 45, pp. 100-117, 1934.

(39) O.L. Reiser, "Modern Science and Non-Aristotelian Logic," The Monist, vol. 46, pp. 299-317, 1936.

(40) N. Rescher, Many-Valued Logic, McGraw-Hill, 1969.

(41) F.S. Roberts, "Tolerance Geometry," Paper P-4430, The Rand Corporation, 1970.

(42) E.S. Santos, "Fuzzy Algorithms," Information and Control, vol. 17, pp. 326-339, 1970.

(43) E.S. Santos, "Probabilistic Grammars and Automata," Information and Control, vol. 21, pp. 27-47, 1972a.

(44) E.S. Santos, "Max-Product Machines," Journal of Mathematical Analysis and Applications, vol. 37, pp. 677-686, 1972b.

(45) E.W. Soja, "The Political Organization of Space," Commission on College Geography, Resource Paper No. 8, Association of American Geographers.

(46) R. Sommer, Personal Space: The Behavioral Basis of Design, Prentice-Hall, 1969.

(47) R.W. Sperry, "Hemisphere Deconnection and Unity of Conscious Awareness, American Psychologist, vol. 23, pp. 723-733, 1968.

(48) P. Suppes, "A Comparison of the Meaning and Uses of Models in Mathematics and the Social Sciences," Synthese, vol. 12, pp. 287-301, 1960.

(49) A. Tarski, "What is Elementary Geometry?" in L. Henkin, P. Suppes, and A. Tarski (eds.), The Axiomatic Method, North-Holland, pp. 16-29, 1959.

(50) R.R. Verma, "Vagueness and the Principle of the Excluded Middle," Mind, vol. LXXIX, pp. 67-77, 1970.

(51) S. Watanabe, Knowing and Guessing, John Wiley, 1970.

(52) S. Watanabe, "Paradigmatic Symbol - A Comparative Study of Human and Artificial Intelligence," IEEE Transactions on Systems, Man and Cybernetics, SMC-4, pp. 100-103, 1974.

(53) W.G. Wee and K.S. Fu, "A Formulation of Fuzzy Automata and its Application as a Model of Learning Systems," IEEE Transactions on Systems Science and Cybernetics, SSC-5, pp. 215-223, 1969.

(54) P. Weiss, "Relativity in Logic," The Monist, vol. 28, pp. 536-548, 1928.

(55) P. Weiss, "On Alternative Logics," The Philosophical Review, vol. 42, pp. 520-525, 1933.

(56) L.A. Zadeh, "Fuzzy Sets," Information and Control, vol. 8, pp. 338-353, 1965.

(57) L.A. Zadeh, "Quantitative Fuzzy Semantics," Information Science, vol. 3, pp. 159-176, 1971.

(58) L.A. Zadeh, "Similarity Relations and Fuzzy Orderings," Information Sciences, vol. 3, pp. 177-200, 1971.

(59) L.A. Zadeh, "Toward a Theory of Fuzzy Systems," in R.E. Kalman and N. DeClaris (eds.), Aspects of Network and System Theory, Holt Rinehart and Winston, pp. 469-490, 1971.

(60) L.A. Zadeh, "A Fuzzy-Set-Theoretic Interpretation of Linguistic Hedges," Memorandum No. ERL-M335, Electronics Research Laboratory, University of California, Berkeley, 1972.

(61) A.A. Zinov'ev, Philosophical Problems of Many-Valued Logic, D. Reidel, 1963.

FREE n-VALUED LUKASIEWICZ ALGEBRAS WITHOUT INVOLUTION

Roberto Cignoli
University of Illinois
at Chicago Circle
Chicago, Illinois

Introduction. The notion of an n-valued Lukasiewicz algebra (n integer ≥ 2) was introduced by G. Moisil in 1941 [10] as a distributive lattice L with 0 and 1 on which there are defined an involutory anti-isomorphism $\sim$ (i.e., L is a DeMorgan algebra, cf. [3]) and n-1 endomorphisms $\sigma_1,\sigma_2,\ldots,\sigma_{n-1}$ that are related with $\sim$ by the formulas:

(1) $$\sigma_i x \vee \sim\sigma_i x = 1$$

and

(2) $$\sim\sigma_i x = \sigma_{n-i}\sim x.$$

In a later work ([11], p. 298) Moisil considered the n-valued Lukasiewicz algebras without involution. These are obtained by omitting the involution $\sim$ in the definition of n-valued Lukasiewicz algebras but adding operators $\bar{\sigma}_1,\ldots,\bar{\sigma}_{n-1}$ that play the role of $-\sigma_i$ in formula (1).

The introduction of these algebras without involution was motivated in part by certain results contained in the present author's thesis (cf. [11], p. 298 and also [2], p. 79) related to the Cartesian product of Lukasiewicz algebras. On the other hand they appear as a natural generalization of Post algebras, where the Lukasiewicz negation $\sim$ is not a primitive operation (cf. [3] and [7]).

Moisil has also introduced the notion of θ-valued Lukasiewicz algebras [11], p. 311 that are, roughly speaking Lukasiewicz algebras without involution in which the ordered set $\{1,2,\ldots,n-1\}$ is replaced by a (possible infinite) bounded totally ordered set J of order type θ. These algebras have been studied in detail by G. Georgescu [9], who has shown that many of the properties of the n-valued Lukasiewicz algebras (as given, for instance, in [3] and [4]) also hold for these more general algebras, and, a fortiori, for the finite valued Lukasiewicz algebras without involution.

In particular Georgescu has proved the existence of the free objects in the category of θ-valued Lukasiewicz algebras, but his results provide no information about the structure of the finitely generated free n-valued Lukasiewicz algebras without involution.

The aim of this paper is to describe the structure of these free algebras because we believe that the knowledge of them may be

This research was partially supported by the Consejo Nacional de Investigaciones de la Republica Argentina.

important not only from the theoretical point of view, but for the possible applications to the theory of switching circuits.

Since the n-valued Lukasiewicz algebras without involution and the n-valued Lukasiewicz algebras share many algebraic properties, we are able to use the same methods we have used in [3] to describe the finitely generated free n-valued Lukasiewicz algebras (cf. also [1]).

In Section 1 we give the precise definitions, and we show that the n-valued Lukasiewicz algebras without involution form an equational class. Moreover, we explore the connections with the P-algebras of G. Epstein and A. Horn [8]. In Section 2 we study the congruence relations, and we characterize the subdirectly irreducibles. This material is used in Section 3, where the free algebras are considered.

1. Definitions and Properties

1.1 Definition ([11]). An n-valued Lukasiewicz algebra without involution (n an integer ≥ 2) is a system $(A,\vee,\wedge,\overline{\sigma}_1,\ldots,\overline{\sigma}_{n-1},0,1)$ such that $(A,\vee,\wedge,0,1)$ is a distributive lattice with zero 0 and unit 1 and $\overline{\sigma}_1,\overline{\sigma}_2,\ldots,\overline{\sigma}_{n-1}$ are unary operators defined on A fulfilling the following axioms:

L1) $\overline{\sigma}_i(x\vee y) = \overline{\sigma}_i x\wedge\overline{\sigma}_i y, \quad \overline{\sigma}_i(x\wedge y) = \overline{\sigma}_i x\vee\overline{\sigma}_i y$

L2) $\overline{\sigma}_i(x\vee\overline{\sigma}_i x) = 0$

L3) $\overline{\sigma}_i(x\wedge\overline{\sigma}_i x) = 1$

L4) $\overline{\sigma}_i x = \overline{\sigma}_i x\vee\overline{\sigma}_{i+1}x \quad (i = 1,2,\ldots,n-2)$

L5) $\overline{\sigma}_i(\overline{\sigma}_j x) = \overline{\sigma}_j(\overline{\sigma}_j x)$

L6) $\overline{\sigma}_i(\overline{\sigma}_j(\overline{\sigma}_k x)) = \overline{\sigma}_k x$

L7) If $\overline{\sigma}_i x\leq\overline{\sigma}_i y$ for $i = 1,2,\ldots,n-1$, then $x\geq y$.

If we define $\sigma_i x = \overline{\sigma}_i(\overline{\sigma}_i x)$ it is not difficult to prove the following properties ([11], p. 298):

L8) $\sigma_i(x\vee y) = \sigma_i x\vee\sigma_i y, \quad \sigma_i(x\wedge y) = \sigma_i x\wedge\sigma_i y$

L9) $\sigma_i x\vee\overline{\sigma}_i x = 1, \quad \sigma_i x\wedge\overline{\sigma}_i x = 0$

L10) $\sigma_i(\sigma_j x) = \sigma_j x$

L11) $\sigma_i x = \sigma_i x\wedge\sigma_{i+1}x \quad (i = 1,2,\ldots,n-2)$

L12) If $\sigma_i x\leq\sigma_i y$ for $i = 1,2,\ldots,n-1$ then $x\leq y$.

In fact, Moisil [11] has proved that the n-valued Lukasiewicz algebras without involution can be characterized as algebras $(A,\vee,\wedge,\sigma_1,\ldots,\sigma_{n-1},\overline{\sigma}_1,\ldots,\overline{\sigma}_{n-1},0,1)$ of type $(2,2,1,1,\ldots,0,0,)$ such that $(A,\vee,\wedge,0,1)$ is a distributive lattice with 0 and 1 and $\sigma_1,\ldots,\sigma_{n-1},\overline{\sigma}_1,\ldots,\overline{\sigma}_n$ satisfy the properties L8) - L12).

The second characterization has the advantage of turning apparent the connections with the n-valued Lukasiewicz algebras, but the first one is more economic.

1.2 Remark. It was proved in [6] that for $n = 3$ the Lukasiewicz negation can be defined in terms of σ_1, σ_2 and the lattice operations, so the three-valued Lukasiewicz algebras without involutions coincide

with the three-valued Lukasiewicz algebras. But if $n \geq 4$ there exist n-valued Lukasiewicz algebras without involution that are not n-valued Lukasiewicz algebras [13].

Our first result is that the class of n-valued Lukasiewicz algebras without involution is equational:

1.3 Theorem. An algebra $(A,\vee,\wedge,\overline{\sigma}_1,\ldots,\overline{\sigma}_{n-1},0,1)$ of type $(2,2,1,1,\ldots,1,0,0)$ is an n-valued Lukasiewicz algebra without involution if and only if $(A,\vee,\wedge,0,1)$ is a distributive lattice with 0 and 1 and $\overline{\sigma}_1,\ldots,\overline{\sigma}_{n-1}$ fulfill the equations L1)-L6) and:

L13) $x \vee \overline{\sigma}_{n-1}x = x, \quad x \vee \overline{\sigma}_1 x = \overline{\sigma}_1 x$

L14) $(x \wedge \overline{\sigma}_i x \wedge \overline{\sigma}_{i+1}(\overline{\sigma}_{i+1}(y))) \vee y = y \quad (i = 1,2,\ldots,n-2).$

Proof. It is essentially the same proof of Theorem 1.4 of [3], and can be omitted.

In what follows A will denote an n-valued Lukasiewicz algebra without involution, and the terms, homomorphisms, subalgebras, etc., will have the usual meaning in universal algebra.

B(A) denotes the Boolean sublattice of all the complemented elements of A. It is easy to see that σ_i maps A into B(A) and that $x \in B(A)$ if and only if $\sigma_i x = x$ (cf. [3],Thm. 9).

L_n denotes the n-valued Lukasiewicz algebra without involution formed by the fractions $j/(n-1)$, $j = 0,1,\ldots,n-1$, with the natural lattice operations and $\overline{\sigma}_i$ defined as follows:

$$\overline{\sigma}_i(j/(n-1)) = \begin{cases} 1 & \text{if } i+j < n \\ 0 & \text{if } i+j \geq n \end{cases} \qquad \begin{matrix} (i = 1,\ldots,n-1, \\ j = 0,1,\ldots,n-1) \end{matrix}$$

Then the $\sigma_i = \overline{\sigma}_i\overline{\sigma}_i$ are given by:

$$\sigma_i(j/(n-1)) = \begin{cases} 0 & \text{if } i+j < n \\ 1 & \text{if } i+j \geq n \end{cases} \qquad \begin{matrix} (i = 1,\ldots,n-1, \\ j = 0,1,\ldots,n-1). \end{matrix}$$

From the definitions of the $\overline{\sigma}_i$ in L_n it follows at once that:

1.4 Theorem. The lattice of subalgebras of L_n is isomorphic to the lattice of all subsets of $\{0,1/(n-1),\ldots,(n-2)/(n-1),1\}$ containing 0 and 1, i.e., the finite Boolean algebra with n-2 atoms. In particular, there are $\binom{n-2}{k-2}$ k-element subalgebras of L_n, $k = 2,3,\ldots,n$.

If we define the binary operation $\Rightarrow$ by the formula:

$$x \Rightarrow y = \sigma_{n-1}x \vee \bigvee_{i=1}^{i=n-2} (\sigma_{n-1}x \wedge \overline{\sigma}_{n-i-1}x \wedge \sigma_{n-i}y) \vee (\sigma_1 x \wedge \sigma_1 y)$$

it is possible to prove that $x \Rightarrow y$ is the greatest element in B(A) satisfying $x \wedge (x \Rightarrow y) \leq y$, and that $(x \Rightarrow y) \vee (y \Rightarrow x) = 1$.

These properties imply that the n-valued Lukasiewicz algebras without involution are P-algebras in the sense of G. Epstein and A. Horn

[8]. Therefore they are Heyting algebras, and the Heyting implication $\rightarrow$ is given by:

$$x \rightarrow y = (x \Rightarrow y) \vee y.$$

This formula for the Heyting implication in n-valued Lukasiewicz algebras was suggested by G. Moisil and proved in the author's thesis (cf. [11], pp. 309-310).

L. Monteiro [12] has characterized the three-valued Lukasiewicz algebras as Heyting algebras with an involution fulfilling the equation:

$$((x \rightarrow z) \rightarrow y) \rightarrow (((y \rightarrow z) \rightarrow y) \rightarrow y) = 1.$$

L. Monteiro's results can be interpreted as a characterization of three-valued Lukasiewicz algebras as a subvariety of P-algebras. The following example shows that in general the n-valued Lukasiewicz algebras without involution do not form a subvariety of the variety of P-algebras: Let $S = \{(0,0),(1/5,2/5),(4/5,3/5),(1,1)\}$. S is a P-subalgebra of $L_6 \times L_6$ but it is not a Lukasiewicz subalgebra.

2. Congruence Relations

A Stone filter of a bounded distributive lattice A is a filter generated by a filter of the Boolean algebra $B(A)$ (cf. [3] and [5]). If A is an n-valued Lukasiewicz algebra without involution it follows that the Stone filters are the filters F with the property that $x \in F$ implies that $\sigma_1 x \in F$. The importance of Stone filters is due to the following:

2.1 Theorem. The correspondence $\theta \longmapsto F_\theta = \{x \in A : x \theta 1\}$ is a lattice isomorphism between the lattice of congruence relations θ on A and the lattice of all Stone filters of A. The inverse isomorphism is given by $F \longmapsto \theta_F$, where $x \theta_F y$ if and only if there is a z in F such that $x \wedge z = y \wedge z$.

2.2 Corollary. The map $\theta \longmapsto F_\theta \cap B(A)$ is an isomorphism between the lattice of congruences of A and the lattice of filters of the Boolean algebra $B(A)$.

If K is a filter of $B(A)$, A/K will denote the quotient algebra of A by the congruence corresponding to K.

We are going to characterize A/K in the case that K is an ultrafilter and in the case that K is a principal filter. We begin by the following result, that can be proved by replacing $-s_i$ by σ_i in the proofs of the corresponding properties for n-valued Lukasiewicz algebras given in [3], Ch. 4.

2.3 Theorem. Let U be an ultrafilter of the Boolean algebra $B(A)$. Then $U_i = \sigma_i^{-1}(U)$, $i = 1,2,\ldots,n-1$ are prime filters of A, and the following relations hold:

(1) $$U = U_i \cap B(A)$$

and

(2) $$U_1 \subseteq U_2 \subseteq \cdots \subseteq U_{n-1}.$$

Conversely, given a prime filter P of A, $P \cap B(A)$ is an ultrafilter of $B(A)$, and $P = (P \cap B(A))_i$ for some i $(1 \leq i \leq n-1)$.

In particular, it follows that the set of prime filters of A, ordered by inclusion, is the cardinal sum of totally ordered sets, each of them having at most $n-1$ elements.

2.4 Corollary. A prime filter P is a Stone filter if and only if $P = (P \cap B(A))_1$, and in this case it is a maximal Stone filter.

If U is an ultrafilter of $B(A)$, define $h_U: A \longrightarrow L_n$ by the following prescription:

$$h_U(x) = (n-j)/(n-1) \text{ if and only if } x \in U_j - U_{j-1},$$
$$j = 1,2,\ldots,n,$$
where $U_0 = \phi$ and $U_n = A$.

A direct computation shows that h_U is an homomorphism and that $h_U(x) = h_U(y)$ if and only if $x \theta_U y$.

From the above remarks it follows at once that:

2.5 Theorem. If U is an ultrafilter of $B(A)$, then A/U is isomorphic to a subalgebra of L_n.

If b is an element in $B(A)$, the principal filter (b) is a Stone filter of A, and A/b will denote the quotient algebra of A by the congruence corresponding to (b), $\theta_{(b)}$.

A/b can be described as follows. Let $A_b = \{x \in A : x \leq b\}$ and for x in A_b define $\overline{\sigma}_i' x = b \wedge \overline{\sigma}_i x$. Then $(A_b, \wedge, \vee, \overline{\sigma}_i', \ldots, \overline{\sigma}_{n-1}', 0, b)$ is an n-valued Lukasiewicz algebra without involution, and the map $h: A \longrightarrow A_b$ given by $h(x) = b \wedge x$ is an epimorphism. Since it is plain that $h(x) = h(y)$ if and only if $x \theta_{(b)} y$, it follows that the algebras A/b and A_b are isomorphic.

2.6 Theorem. If $b_1, b_2, \ldots, b_k$ are elements of $B(A)$ such that $b_1 \vee b_2 \vee \cdots \vee b_k = 1$ and $b_i \wedge \bigvee_{j \neq i} b_j = 0$, then:

$$A \cong A_{b_1} \times A_{b_2} \times \cdots \times A_{b_k} \cong A/b_1 \times A/b_2 \times \cdots \times A/b_k$$

Proof. By a well-known result of G. Birkhoff, the map $x \longmapsto (b_1 \wedge x, \ldots, b_k \wedge x)$ is a lattice isomorphism from A into the direct product $A_{b_1} \times \cdots \times A_{b_k}$, and it is easy to check that it is also a Lukasiewicz isomorphism.

2.7 Corollary. If $B(A)$ is finite and it has k atoms, then A is isomorphic to the direct product of k subalgebras of L_n.

Proof. If $a_1, \ldots, a_k$ are the atoms of $B(A)$, $A = A/a_1 \times \cdots \times A/a_k$, and since the atoms of $B(A)$ are in correspondence with the ultrafilters,

it follows from Theorem 2.5 that each A/a_i is isomorphic to a subalgebra of L_n.

The following characterization of the subdirectly irreducible algebras can now be easily proved by using the previous results of this section:

2.8 Theorem. For any n-valued Lukasiewicz algebra without involution A with more than one element the following properties are equivalent:

(i) $B(A) = \{0,1\}$.
(ii) A is totally ordered.
(iii) The prime filters of A form a chain.
(iv) A is isomorphic to a subalgebra of L_n.
(v) A is simple.
(vi) A is subdirectly irreducible.
(vii) A is irreducible.

2.9 Corollary. Every n-valued Lukasiewicz algebra without involution with more than one element is a subdirect product of a family of subalgebras of L_n.

3. Free Algebras

3.1 Definition. If c is a cardinal number > 0, then by a free n-valued Lukasiewicz algebra without involution with c free generators we mean any n-valued Lukasiewicz algebra without involution $F_n(c)$ such that:

1) $F_n(c)$ has a set of generators G of power c, and
2) Any map f from G into any n-valued Lukasiewicz algebra without involution A can be extended to a homomorphism h_f from $F_n(c)$ into A.

Since n-valued Lukasiewicz algebras without involution form a variety, it follows from a well-known theorem of G. Birkhoff that for any cardinal $c > 0$ there exists $F_n(c)$, and it is unique up to isomorphisms. Moreover, the homomorphism h_f that appears in Definition 2.1 is unique.

In particular, there is a one-to-one correspondence between the functions from G into L_n and the homomorphisms from $F_n(c)$ into L_n. If G is finite, there are finitely many functions from G into L_n; therefore, from Theorem 2.8 it follows at once that the finitely generated free algebras are finite. Moreover it is easy to check that if A is a subalgebra of L_n, in order that $h_f(F_n(c)) = A$ it is necessary and sufficient that $f(G) \subseteq A$ and $f(G) \not\subseteq A'$ for any maximal proper subalgebra A' of A (cf. [3]. Lemma 7.5).

Assume now that r is a finite cardinal > 0, and $G = \{g_1,\ldots,g_r\}$ a set of generators of $F_n(r)$. Let $F(n,r)$ denote the set of all functions $f:G \longrightarrow L_n$, and for $f \in F(n,r)$ let U_f be the ultrafilter of $B(F_n(r))$ corresponding to the homomorphism h_f. Since $F_n(r)$ is finite by Theorem 2.6 it follows that:

$$F_n(r) \cong \prod_{f \in F(n,r)} F_n(r)/U_f.$$

If A is a subalgebra of L_n with k elements $(2 \leq k \leq n)$ then there are $k-2$ distinct maximal proper subalgebras of A, and the intersection of s of these subalgebras will be a subalgebra with $k-s$ elements. Now we can apply the same counting arguments as in ([3], Chap. 7, case n even) to obtain the structure of $F_n(r)$. (cf. also [1]).

3.2 Theorem. Let A_{kj}, $j = 1,2,\ldots,\binom{n-2}{k-2}$; $k = 2,3,\ldots,n$ denote the k-element subalgebras of L_n, and let

$$a(r,k) = \sum_{s=0}^{k-2} (-1)^s \binom{k-2}{s} (k-s)^r.$$

Then:

$$F_n(r) \cong \prod_{k=2}^{n} \Big(\prod_{j=1}^{\binom{n-2}{k-2}} A_{kj}^{a(r,k)} \Big)$$

and

$$|F_n(r)| = \prod_{k=2}^{n} k^{\binom{n-2}{k-2} a(r,k)}$$

where $|F_n(r)|$ denotes the number of elements of $F_n(r)$.

REFERENCES

1. J. Berman, Algebras with modular lattice reducts and simple subdirectly irreducibles, Discrete Math. 11 (1975), 1-8.

2. V. Boicescu, Sur les algebres de Lukasiewicz, In: Logique-Automatique-Informatique, Ed. Acad, Rep. Soc. Roumamie, Bucharest, 1971, 71-89.

3. R. Cignoli, Moisil algebras, Notas de Lógica Matemática No. 27, Univ. Nac. del Sur, Bahia Blanaa, 1970, 47 pp.

4. R. Cignoli, Representation of Lukasiewicz and Post algebras by continuous functions, Colloq. Math. 24 (1972), 127-138.

5. R. Cignoli, Stone filters and ideals in distributive lattices, Bull. Math. Soc. Scie. Math. R.S. Roum. 15 (63) (1971), 131-137.

6. R. Cignoli and A. Monteiro, Boolean elements in Lukasiewicz algebras II, Proc. Japan Acad. 41 (1965), 676-680.

7. G. Epstein, The lattice theory of Post algebras, Trans. Amer. Math. Soc. 95 (1960), 300-317.

8. G. Epstein and A. Horn, P-algebras, an abstraction from Post algebras, Algebra Universalis 4 (1974), 195-206.

9. G. Georgescu, Les algebres de Lukasiewicz θ-valentes, In: Logique, Automatique, Informatique, Ed. Acad. Rep. Soc. Roumanie, Bucharest, 1971, 99-169.

10. G. Moisil, Notes sur les logiques non-chrysippiennes, Ann. Sci. Univ. Jassy, 27 (1941), 86-98. This paper has been reproduced in the book [9], pp. 238-243.

11. G. Moisil, Essais sur les logiques non chrysippiennes, Ed. Acad. Rep. Soc. Roumanie, Bucharest, 1972.

12. L. Monteiro, Les algebres de Heyting et de Lukasiewicz trivalentes, Notre Dame J. Formal Logic 11 (1970), 453-466.

13. W. Suchon, Inequivalence des deux definitions des algebres de Lukasiewicz, Zeszyty Naukowe UJ, Prace z Logiki 7 (1972), 31-34.

Department of Mathematics
University of Illinois at Chicago Circle
Chicago, Iliinois

and

Departamento de Matemática
Universidad Nacional del Sur
Bahia Blanca, Argentina

ON THE ALGEBRAS CORRESPONDING TO THE n-VALUED ŁUKASIEWICZ-TARSKI LOGICAL SYSTEMS

R. Grigolia
Tbilisi University
U S S R

A class of algebras corresponding to n-valued logical systems of Łukasiewicz-Tarski $L_n (2 < n < \aleph_0)$ [5] is examined in this work. These algebras are called MV_n-algebras [4]. The class of all MV_n-algebras is a special subclass of all MV-algebras, which was defined by Chang in [2] as corresponding to the $\aleph_0$-valued propositional calculi of Łukasiewicz $L_{\aleph_0}$.

All the material consists of five sections. Axioms of MV_n-algebras and important consequences, and the representation's theorem for MV_n-algebras are given in the first section.

An axiomatization of the logical systems L_n in terms of Łukasiewiczian implication ($\supset$) and negation ($\sim$) and $+$, $\cdot$ ($\alpha + \beta \overset{\text{def}}{=} \sim\alpha \supset \beta$, $\alpha \cdot \beta \overset{\text{def}}{=} \sim(\alpha \supset \sim\beta)$) is suggested in the second section.

A lattice of equational subclasses of the equational class of MV-algebras generated by finite MV-algebras, in other words a lattice of all extensions with the finite model property of $L_{\aleph_0}$, is described in the third section.

A description of the free MV_n-algebras with m generators, i.e., a structure of non-equivalent in L_n formulae of m variables, is given in the fourth section.

I. An MV_n-algebra is a system $\langle A, +, \cdot, {}^-, 0, 1\rangle$ where A is a non-empty set of elements, 0 and 1 are distinct constant elements of A, $+$ and $\cdot$ are binary operations on elements of A, and $^-$ is an unary operation on elements of A obeying the following axioms [4].

Axiom 1	$x + y = y + x$	$x \cdot y = y \cdot x$
Axiom 2	$x + (y+z) = (x+y) + z$	$x \cdot (y \cdot z) = (x \cdot y) \cdot z$
Axiom 3	$x + \bar{x} = 1$	$x \cdot \bar{x} = 0$
Axiom 4	$x + 1 = 1$	$x \cdot 0 = 0$
Axiom 5	$x + 0 = x$	$x \cdot 1 = x$
Axiom 6	$[x+y]^- = \bar{x} \cdot \bar{y}$	$[x \cdot y]^- = \bar{x} + \bar{y}$
Axiom 7	$x = [x]^-$	
Axiom 8	$\bar{0} = 1$	
Axiom 9	$x \vee y = y \vee x$	$x \wedge y = y \wedge x$
Axiom 10	$x \vee (y \vee z) = (x \vee y) \vee z$	$x \wedge (y \wedge z) = (x \wedge y) \wedge z$

Axiom 11 $x + (y \wedge z) = (x+y) \wedge (x+z)$ $\qquad x \cdot (y \vee z) = (x \cdot y) \vee (x \cdot z)$

Axiom 12 $(n-1)x + x = (n-1)x$ $\qquad x^{n-1} \cdot x = x^{n-1}$

If $n \geq 3$, the following axiom is added

Axiom 13 $[(jx) \cdot (\bar{x} + [(j-1)x]^{-})]^{n-1} = 0$ $\qquad [(n-1)[x^{j} + (\bar{x} \cdot [x^{j-1}]^{-})] = 1$

where $1 < j < n-1$ and j does not divide $n-1$; $x \vee y = (x \cdot \bar{y}) + y$, $x \wedge y = (x + \bar{y}) \cdot y$, $0 \cdot x = 0$ and $(m+1)x = mx + x$, $x^{0} = 1$ and $x^{m+1} = x^{m} \cdot x$.

An algebra obeying the Axioms 1-11 is MV-algebra by Chang [2].

We define a partial ordering relation $\leq$ on A : $x \leq y$ if and only if $x \vee y = y$.

The example of an MV_n-algebra is the algebra

$$\langle \mathcal{L}_n , +, \cdot , {}^{-}, 0, 1 \rangle ,$$

where

$$\mathcal{L}_n = \{0 , 1/n-1 , \ldots , n-2/n-1 , 1\} ,$$

$$x + y = \min(1 , x+y) ,$$

$$x \cdot y = \max(0 , x+y-1) ,$$

$$\bar{x} = 1 - x .$$

$\mathcal{L}_m (m \leq n)$ is MV_n-algebra if and only if $m-1$ divides $n-1$ [4].

A subset F of A is a multiplicative filter (M-filter) [2] if and only if

(1) $1 \in F$;

(2) if $x, y \in F$, then $x \cdot y \in F$;

(3) if $x \in F$ and $y \geq x$, then $y \in F$.

Theorem 1 The lattice of M-filters on A is isomorphic to the lattice of congruence relations over A .

The M-filter F is said to be proper, if $F \neq A$. M-filter F is maximal if and only if F is a proper M-filter, and whenever F' is an M-filter such that $F \subseteq F' \subseteq A$, then either $F = F'$ or $F' = A$. M-filter F is prime provided it is proper and the condition $x \vee y \in F$ implies that either $x \in F$ or $y \in F$.

Theorem 2 Let F is a proper M-filter of A , then the following conditions are equivalent:

(1) F is a maximal filter ;

(2) F is prime ;

(3) for every $x \in A$ either $x \in F$ or $\sim x = \sim(x^{n-1}) \in F$.

A/F is a quotient algebra of A and ρ , where ρ is the unique congruence relation associated with F .

Theorem 3 For any MV_n-algebra A, A/F is isomorphic to algebra $\mathcal{L}_m$, where F is a maximal M-filter and $m-1$ divides $n-1$.

Theorem 4 Any (finite) MV_n-algebra A is isomorphic to the subdirect (direct) product of the algebras $\mathcal{L}_m$, where $m \leq n$ and $m-1$ divide $n-1$.

Note that the same result for 3-valued Łukasiewicz algebras was received by L. L. Esakia in [3]. MV_k-algebras $\langle V, +, \cdot, \vee, \wedge, \bar{}, 0, 1, \delta_1, \ldots, \delta_{k-1} \rangle$ which also correspond to k-valued logical systems L_k, were independently introduced by G. Malinowski in [6]. Every MV_k-algebra is isomorphic with the subdirect product algebras $\langle \mathcal{L}_z, +, \cdot, \vee, \wedge, \bar{}, 0, 1, \delta_1, \ldots, \delta_{k-1} \rangle$ where $z \leq k$,

$$\delta_i\left(\frac{j}{k-1}\right) = \begin{cases} 1 & \text{for } 1 \leq i \leq j+1 \\ 0 & \text{for } j+1 \leq i \leq k-1 \end{cases};$$

if k is even, then z is even too [6].

II. The formulae F_L of the logical systems L_n are built up of denumerably many propositional variables with three logical operators $+$, $\cdot$, and $\sim$. Łukasiewicz-Tarski's n-valued logical system L_n is the set of all formulae which are satisfied by the algebra $\langle \mathcal{L}_n, +, \cdot, \bar{}, 0, 1 \rangle$.

The following axiom schemas are the axiom for calculi L_n:

L_n1 $\alpha \supset (\beta \supset \alpha)$

L_n2 $(\alpha \supset \beta) \supset ((\beta \supset \gamma) \supset (\alpha \supset \gamma))$

L_n3 $(\sim\alpha \supset \sim\beta) \supset (\beta \supset \alpha)$

L_n4 $((\alpha \supset \beta) \supset \beta) \supset ((\beta \supset \alpha) \supset \alpha)$

If $n > 3$, the following axiom is added:

L_n6 $(n-1)((\sim\alpha)^j + (\alpha \cdot (j-1)\alpha))$

where $1 < j < n-1$ and j does not divide $n-1$; $\alpha \supset \beta \overset{\text{def}}{=} \sim\alpha + \beta$, $m\alpha$ and α^m are abbreviations for $\alpha + \ldots + \alpha$ and $\alpha \cdot \ldots \cdot \alpha$, respectively. The unique rule of detachment is modus ponens. The completeness of this axiom is proved by the representation's theorem for MV_n-algebras.

III. Let $\mathcal{M}$ be the lattice of equational subclasses of the equational class of all MV-algebras, generated by finite MV-algebras.

Z is the set of all non-negative integers. By means of $\mathcal{N}$ we'll denote the set of all finite sequences $\{a_1, \ldots, a_n\}$ where $a_1, \ldots, a_n \in Z$ are such that for any a_i and a_j $(i \neq j)$, a_i does not divide a_j. The relation $\leq$ is defined over $\mathcal{N}$. For any $a, b \in \mathcal{N}$ $a \leq b$ iff every element of a divides some element of b. $\leq$ is a partial ordering relation. Elements of $\mathcal{N}$ are said to be the crowns.

Let $\{m_1, \ldots, m_n\}$ be any finite sequence of elements of Z. If m_j divides $m_i (i \neq j)$, then we'll cross out m_j in this sequence. Finally, we'll get a crown. We'll denote such operation on the sequence of elements of Z by $*$.

On the ordered set $\mathcal{N}$ the least upper bound $(\vee)$ and the greatest lower bound $(\wedge)$ of elements $a = \{a_1, \ldots, a_n\}$ and $b = \{b_1, \ldots, b_m\}$ are defined as follows:

$$a \vee b = \{a_1, \dots, a_n, b_1, \dots, b_m\}^*$$

$$a \wedge b = \{g.c.d.(a_1, b_1), g.c.d.(a_1, b_2), \dots, g.c.d.(a_n, b_m)\}^*$$

where g.c.d. is the greatest common divisor.

Theorem 4 The lattice $\mathcal{M}$ is isomorphic to the lattice $\mathcal{N}$.

IV. We'll denote a free algebra in the class of MV_n-algebras with m generators by $\mathfrak{U}_m^{(n)}$.

Let's write out the increasing sequence of numbers $n_1, \dots, n_k$ where $n_1 - 1, \dots, n_k - 1$ are all divisors of $n - 1$ $(n_1 = 2, n_k = n)$. Let's define the sequence

$$v_{n_1}, \dots, v_{n_k}$$

as follows: $v_{n_1} = 2^m$, $v_{n_2} = n_2^m - 2^m$. Let all v_{n_i} be defined for $i < j$. Let's define v_{n_j} .

$$v_{n_j} = n_j^m - (v_{n_{j_1}} + \dots + v_{n_{j_e}})$$

where $n_{j_1} - 1, \dots, n_{j_e} - 1$ are all divisors of the number $n_j - 1$ distinguished from $n_j - 1$.

Theorem 5 The algebra $\mathcal{L}_{n_1}^{v_{n_1}} \times \dots \times \mathcal{L}_{n_k}^{v_{n_k}}$ is isomorphic to $\mathfrak{U}_m^{(n)}$.

V. Axiomatization of 3-valued Łukaciewicz algebras in terms of pseudo-Boolean operations $\vee$, $\wedge$, $\rightarrow$, $\neg$, 1 and involution $\sim$ is given in [7]. The algebra satisfying these axioms is said to be simply Łukasiewicz algebra. We'll give the axiomatization of Łuksiewicz algebras in terms $\vee$, $\wedge$, $\rightarrow$, $\neg$, 0, 1 and $\sim$, which is obtained from axiomatization of L. Monteiro by adding of two axioms. More precisely, Łukasiewicz algebra is a system $\langle A, \vee, \wedge, \rightarrow, \neg, \sim, 0, 1 \rangle$ obeying the following axioms:

1 $x \rightarrow x = 1$

2 $x \wedge (x \rightarrow y) = x \wedge y$

3 $x \rightarrow (y \wedge z) = (x \rightarrow y) \wedge (x \rightarrow z)$

4 $(x \vee y) \rightarrow z = (x \rightarrow z) \wedge (y \rightarrow z)$

5 $((x \rightarrow z) \rightarrow y) \wedge ((y \rightarrow x) \rightarrow y) = y$

6 $\sim\sim x = x$

7 $\sim(x \wedge y) = \sim x \vee \sim y$

8 $x \wedge \sim x = (x \wedge \sim x) \wedge (y \vee \sim y)$

9 $\sim 1 = 0$

10 $x \rightarrow 0 = \neg x$

The example of Łukasiewicz algebra is the algebra

$$\langle \mathcal{L}_n', \vee, \wedge, \rightarrow, \neg, \sim, 1, 0 \rangle$$

where $\mathcal{L}_n' = \{0, 1/n-1, \ldots, n-2/n-1, 1\}$,

$x \vee y = \max(x, y)$,

$x \wedge y = \min(x, y)$,

$$x \rightarrow y = \begin{cases} 1 & \text{for } x \leq y \\ y & \text{for } x > y \end{cases} ,$$

$$\neg x = \begin{cases} 1 & \text{for } x = 0 \\ 0 & \text{for } x \neq 0 \end{cases} ,$$

$\sim x = 1 - x$.

It's easy to see that algebras $\mathcal{L}_n$ and $\mathcal{L}_n'$ are functionally equivalent.

We'll introduce the following notations.

$\mathfrak{X}$ is the set of all prime filters of algebra A ;

h(a) is the set of all such $\varphi \in \mathfrak{X}$ that $a \in \varphi$;

$P(\mathfrak{X})$ is the class of all h(a) for $a \in A$.

Let $g(\varphi) = A - \tilde{\varphi}$ be for every $\varphi \in \mathfrak{X}$, where $\tilde{\varphi}$ is the set of all $\sim a$ such that $a \in \varphi$. Let $\sim\chi = \mathfrak{X} - g(\chi)$ be for any $\chi \in P(\mathfrak{X})$.

h is an isomorphism $\langle A, \vee, \wedge, \sim \rangle$ onto $\langle P(\mathfrak{X}), \cup, \cap, \sim \rangle$ [1]. A pair $\langle \mathfrak{X}, R \rangle$, where $\mathfrak{X}$ is a nonempty set with a quasiordering relation R over it is said to be Kripke model. A triple $\langle \mathfrak{X}, R, g \rangle$, where $\langle \mathfrak{X}, R \rangle$ is called Kripke model, and g is involution on $\mathfrak{X}$, is Kripke model with involution.

Theorem 6 Let B(A) be the set of all the elements $\Delta a (= \sim \neg \neg \sim a)$, where $a \in A$. Then B(A) is closed under the primitive operations of algebra A . Moreover the system $\langle B(A), \vee, \wedge, \sim \rangle$ is a Boolean algebra and in B(A) is true equality $\sim x = \neg x$.

Lemma 1 For any Łukasiewicz algebra A the following conditions are equivalent:

(i) $\varphi_1 \cap B(A) \subseteq \varphi_2 \cap B(A)$,

(ii) for any $a \in A$, $a \in \varphi_2 \Rightarrow \nabla a = \neg\neg a \in \varphi_1$,

(iii) for any $a \in A$, $\Delta a \in \varphi_1 \Rightarrow a \in \varphi_2$,

for any $\varphi_1, \varphi_2 \in \mathfrak{X}$.

Let's define the quasiordering relation R over $\mathfrak{X}$:

$R(\varphi_2, \varphi_1)$ iff $\varphi_1 \cap B(A) \subseteq \varphi_2 \cap B(A)$.

For any $\chi \subset \mathfrak{X}$, $R(\chi)$ is the set of all $\varphi \in \mathfrak{X}$ such that $R(\varphi', \varphi)$ where $\varphi' \in \mathfrak{X}$.

Lemma 2 $h(\nabla a) = R(h(a))$.

In Łukasiewicz algebra the equality $x \rightarrow y = \Delta \sim x \vee y \vee (\nabla \sim x \wedge \nabla y)$ is true [7].

Let's define the operations $\rightarrow$, $\neg$ on $P(\mathcal{X})$ as follows:

$$x \rightarrow y = \sim R(x) \cup y \cup (R(\sim x) \cap R(y)) , \quad \neg x = \sim R(x) .$$

$\langle P(\mathcal{X}), \cup, \cap, \rightarrow, \neg, \sim, \mathcal{X}, \emptyset \rangle$ is Łukasiewicz algebra. We'll call the Łukasiewicz algebra of this kind the Brouwerian quasifield of sets of Kripke model with involution.

Theorem 7 Every Łukasiewicz algebra is isomorphic to the Brouwerian quasifield of sets of the suitable Kripke model with involution.

REFERENCES

[1] A. Bialynicky-Birula and H. Rasiowa, "On the representation of quasi-Boolean algebras", Bull. Acad. Pol., cl. III, 5, pp. 259-261, 1957.

[2] C. C. Chang, "Algebraic analysis of many-valued logics", Trans. Amer. Math. Soc., 88, pp. 467-490, 1958.

[3] L. L. Esakia, "On Łuksiewiczian algebras", IVth International Congress for Logic, Methodology and Philosophy of Science, Abstracts, Bucharest, p. 22, 1971.

[4] R. S. Grigolia, "Algebraic analysis of n-valued Łukasiewicz-Tarski logical systems", Proc. Tbilisi. Univ. A6-7 (149-150), pp. 121-132, 1973.

[5] J. Łukasiewicz and A. Tarski, "Untersuchungen über den aussagenkalkül", Comptes Rendus de la Societé de Sciences et de Lettres de Vasovie, cl. III, 23, pp. 30-50, 1930.

[6] G. Malinowski, "MV_k-algebras", Bull of the Sect. on Logic, No. 3, 2, pp. 185-191, 1973.

[7] L. Monteiro, "Les algebras de Heyting et de Łukasiewicz trivalentes", Notre Dame Form. Log., no. 4, 11, 1970.

A THEOREM ON THE FINITENESS OF THE DEGREE OF MAXIMALITY OF THE n-VALUED ŁUKASIEWICZ LOGIC

Ryszard Wójcicki
Institute of Philosophy and Sociology
Polish Academy of Science
Poland

§1. The objective of the paper

By the language of a Łukasiewicz propositional logic I shall understand the algebra of formulas

(1) $$L = (L, \Rightarrow, \vee, \wedge, \neg)$$

formed by means of propositional variable $p, q, r, \ldots$ and the standard connectives.

A logical matrix for L is defined to be a pair

(2) $$M = (A, I),$$

where A is an algebra similar to L, and the subset $I \subseteq A$ is the set of designated elements of A. The operations of A corresponding to the operations (connectives) of L will be denoted again by $\Rightarrow, \vee, \wedge, \neg$ respectively. A homomorphism h from L into A will be called a valuation of formulas of L in M. We shall say that h verifies α, provided that $h\alpha \in I$. A formula α will be said to be a M-consequence of a set of formulas X, in symbols

(3) $$\alpha \in Cn_M(X),$$

iff for each valuation h, h verifies α whenever h verifies all formulas in X.

If for every a, b in A, $a \Rightarrow a = b \Rightarrow b$, the element $a \Rightarrow a$ will be called the unit of A and it will be denoted by 1_A. If 1_A exists, the algebra A will be called a matrix algebra for L. Whenever I shall refer to an algebra A in a context in which the notion of a matrix ought to be used, it will be assumed

that A is a matrix algebra, and the symbol A stands for $(A,\{1_A\})$. For example: Cn_A will abbreviate $Cn_{(A,\{1_A\})}$.

The n-valued <u>Łukasiewicz matrix</u> M_n will be assumed to be the algebra formed by the set of fractions 0, 1/n-1, 2/n-1,...,1, and the familiar operations $\Rightarrow$, $\vee$, $\wedge$, $\neg$. Observe that $1 = a \Rightarrow a$, for every a in M_n, and thus 1 is the unit of the algebra (the designated element of the matrix) M_n.

Denote the consequence operation Cn_{M_n} by Cn_n for short. In what follows, by the n-valued <u>Łukasiewicz propositional logic</u> I shall understand the pair

(4) $$Ł_n = (L, Cn_n).$$

PROPOSITION 1 (cf./1/) $\alpha \in Cn_n(X)$ iff α is derivable from $X \subseteq Cn_n(\emptyset)$ by Modus Ponens.

In view of Proposition 1, $Ł_n$ can be formalized by selecting $Cn_n(\emptyset)$ to be the set of axioms for $Ł_n$, and selecting Modus Ponens to be the sole rule of inference of $Ł_n$. Observe that $Cn_n(\emptyset)$ consists of exactly those formulas which are verified by all valuations in M_n. It is also easily seen to be closed under substitutions (endomorphisms of L).

By enlarging the set of axioms of $Ł_n$ by some new ones such that the resulting set is again closed under substitutions, we define a so-called <u>structural</u> (or <u>normal</u>) <u>axiomatic strengthening</u> of $Ł_n$ (of Cn_n). More precisely: a consequence operations C defined on L is said to be a strengthening of Cn_n, $Cn_n \leqslant C$, iff $Cn_n(X) \subseteq C(X)$ for every $X \subseteq L$. If for a fixed Z, $C(X) = Cn_n(X \cup Z)$ for every X, C is said to be an <u>axiomatic strengthening</u> of Cn_n. Finally, C is said to be <u>structural</u> iff $\alpha \in C(X)$ implies that $e\alpha \in C(eX)$, for each endomorphism e of L. The cardinality of all structural axiomatic strengthenings of $Ł_n$ is said to be the (<u>cardinal</u>) <u>degree of completeness</u> of $Ł_n$. As known the degree of completeness of $Ł_n$ is finite for all finite n.

Clearly, one may also strengthen $Ł_n$ by adding new inference rules. The cardinality of the set of all structural strengthenings C of Cn_n will be called

the degree of maximality of $Ł_n$, $dm(Ł_n)$. Obviously, $dm(Ł_n)$ is at least as great as the degree of completeness of $Ł_n$. As a matter of fact, for all finite n, $dm(Ł_n)$ is greater than the corresponding degree of completeness, but the argument showing that is a bit more involved.

The objective of this paper is to prove the following theorem.

THEOREM. For all finite $n \geqslant 2$, $dm(Ł_n)$ is finite.

(Note. The notion of a M-consequence, a matrix consequence, was defined by Łoś and Suszko. Some theorems, relevant to this paper, on matrix consequence were established in /2/, /3/ and /4/. A definition of the degree of maximality was proposed by the author of this paper in /5/. For some results concerning $dm(Ł_n)$ see /5/ and /6/.)

§2. $Ł_n$-algebras

Given a consequence C defined on L, denote by Matr(C), the set of all matrices M such that $C \leqq Cn_M$.

LEMMA 1. (cf./4/) Each structural consequence C is uniquely defined by Matr(C), i.e. for any two structural consequences C, C′ defined on L, $C = C'$ only if $Matr(C) = Matr(C')$.

As an immediate consequence of Lemma 1 we have:

(I) $$dm(Ł_n) \leqslant \text{card}\{\Theta: \Theta \subseteq Matr(Cn_n)\}.$$

Define two matrices M, N to be equivalent, $M \sim N$, provided that $Cn_M = Cn_N$. Clearly, (I) implies:

(II) $$dm(Ł_n) \leqslant \text{card}\{\Theta: \Theta \subseteq Matr(Cn_n)/_{\sim}\}$$

For each matrix $M = (A,I)$ for L, define $a \approx_M b$ $(a,b \in A)$ to hold true iff $a \Rightarrow b$, $b \Rightarrow a \in I$.

LEMMA 2. If $M = (A,I)$ is in $Matr(Cn_n)$, the following conditions are

valid:

i. The relation $\approx_M$ is a congruence of A.

ii. If $a \in I$, then $I = |a|_M$, where $|a|_M = \{b: b \approx_M a\}$.

iii. I is the unit of the quotient algebra $A/\approx_M$.

iv. $M \sim A/\approx_M$, i.e. the matrices M and $(A/\approx_M, I)$ are equivalent.

Proof. It is an easy matter to verify i. and ii. One may also easily see that $a \Rightarrow a$ is in I, for each a. Thus, by ii., $I = |a \Rightarrow a|_M = |a|_M \Rightarrow |a|_M$, which proves iii. Now, observe that i. and ii. taken togather state that $\approx_M$ is so-called matrix congruence, which (cf./4/) yields iv. concluding the proof.

Denote by $Alg^R(Cn_n)$ the set of all algebras of the form $A/\approx_M$, where $M \in Matr(Cn_n)$. We may improve (II) as follows:

(III) $$dm(Ł_n) \leqslant card\{\Theta: \Theta \subseteq Alg^R(Cn_n)/\sim\}.$$

For the notions involved in the next lemma consult Rasiowa /7/.

LEMMA 3. Each $Ł_n$ is a calculus of the class S (i.e. a standard, implicative, extensional propositional calculus) and $Alg^R(Cn_n)$ is the class of $Ł_n$-algebras.

Proof. One may easily verify that each $Ł_n$ satisfies the conditions (s_1) - (s_8) put by Rasiowa (cf./7/,p.179) on calculi in S. It is also a routine matter to check that the conditions (a_1) - (a_4) (cf./7/p.181), which define the notion of a S-algebra, hold true for each A in $Alg^R(Cn_n)$. To see that each $Ł_n$-algebra is in $Alg^R(Cn_n)$, note first that in virtue of (a_1), (a_2) each such algebra A is in $Matr(Cn_n)$. In turn, (a_4) implies that $\approx_A$ is the identity relation and thus A is in $Alg^R(Cn_n)$, which concludes the proof.

Denote by $HSP(M_n)$ (by $SP(M_n)$) the least class of algebras containing the algebra M_n and closed under the operations of forming direct products,

subalgebras, and homomorphic immages (direct product and subalgebras). It is easily seen that each algebra $A \in HSP(M_n)$ possesses the unit element 1_A.

LEMMA 4. $Alg^R(Cn_n) = HSP(M_n)$

Proof. We shall need the following identities:

(i_1) $\qquad 1_A \Rightarrow a = a$

(i_2) $\qquad ((a \Rightarrow b) \Rightarrow b) \wedge a = a$

(i_3) $\qquad a \wedge b = b \wedge a$

which holds true in each A in $HSP(M_n)$ Indeed they hold true in each M_n and, since the operations involved in forming $HSP(M_n)$ preserve identities, they are valid in all algebras in $HSP(M_n)$.

M_n is easily seen to be an L_n-algebra and thus to prove that $HSP(M_n) \subseteq Alg^R(Cn_n)$ it is enough to prove that the intersection of the two sets of algebras is closed under the operations of forming algebras, direct product and homomorphic immages. Only the last case is not trivial. Assume then that A is both in $HSP(M_n)$ and in $Alg^R(Cn_n)$. Let for some homomorphism h, $B = hA$. We easily see that $Cn_n(\emptyset) \subseteq Cn_A(\emptyset) \subseteq Cn_B(\emptyset)$, which proves that B satisfies the condition (a_1) for L_n-algebras. Here and in the sequel we make use of Lemma 3 and the conditions $(a_1) - (a_4)$ Stated for L_n-algebras in /7/ p.181.

In view of Proposition 1, in order to establish (a_2) we may confine our considerations to Modus Ponens. Suppose that for some formulas α, β and a valuation h in B, $h(\alpha \Rightarrow \beta) = h\alpha = 1_B$. We have $h(\alpha \Rightarrow \beta) = h\alpha \Rightarrow h\beta = 1_B \Rightarrow h\beta$, and since B is in $HSP(M_n)$, by (i_1) we arrive to $1_B \Rightarrow h\beta = h\beta$. Combining all these identities we get $h\beta = 1_B$, as it is desired.

We have proved that $Cn_n \leqslant Cn_B$. This in particular entails that $\alpha \Rightarrow \beta \in Cn_B(\alpha \Rightarrow \beta, \beta \Rightarrow \alpha)$, for all α, β, γ, which, in turn, allows us to prove (a_3).

Now assume that for some a,b in B, $a \Rightarrow b = b \Rightarrow a = 1_B$. With the help of (i_2) we have $((a \Rightarrow b) \Rightarrow b) \wedge a = a$ and similarly $((b \Rightarrow a) \Rightarrow a) \wedge b = b$,

and thus we have also $(1_B \Rightarrow b) \wedge a = a$ and $(1_B \Rightarrow a) \wedge b = b$. Applying (i_1) we arrive to $b \wedge a = a$ and $a \wedge b = b$, which by (i_3) gives us finally $a = b$, establishing (a_4). Thus we have proved that B is an $Ł_n$-algebra or equivalently that B is in $Alg^R(Cn_n)$, concluding this part of the proof.

Define $\alpha \approx \beta$ to hold true whenever both $\alpha \Rightarrow \beta$ and $\beta \Rightarrow \alpha$ are in $Cn_n(\emptyset)$. The relation $\approx$ is a congruence of the language L of $Ł_n$, and the quotient $L/_{\approx}$ of L is known as the Lindenbaum algebra of $Ł_n$. Observe that $\|\alpha \Rightarrow \alpha\| = \{\beta : \beta \approx \alpha \Rightarrow \alpha\}$ is the unit of $L/_{\approx}$. For the purpose of the proof we shall show that $L/_{\approx}$ is in $HSP(M_n)$.

Let $\{h_t : t \in T\}$ be the set of all valuations in M_n. Form the power $(M_n)^T$. Let h be the valuation in $(M_n)^T$ such that for each propositional variable p and for each t in T, $(hp)_t = h_t p$, where $(hp)_t$ denotes the projection of hp onto the exist. Since h is a homomorphism, the set $\{h\alpha : \alpha \in L\}$ is a subalgebra of $(M_n)^T$.

Now, verify that $\alpha \approx \beta$ iff for each h_t, $\{h_t\alpha = h_t\beta\}$, or equivalently $h\alpha = h\beta$. It follows that $L/_{\approx}$ is isomorphic with $(M_n)^T$, which concludes the argument showing that $L/_{\approx}$ belongs to $HSP(M_n)$.

On the other hand, as a particular instance of some general results established by Rasiowa (cf./7/, Chapter VIII, Theorems 6.5 and 6.7), we have that $L/_{\approx}$ is an Ł-algebra, and moreover it is free in the class of all Ł-algebras.

These facts allow us to prove that all equations valid for elements of algebras in $HSP(M_n)$ are valid for elements of each $Ł_n$-algebra. Since $HSP(M_n)$ is equationally definable then it follows that all $Ł_n$-algebras are in $HSP(M_n)$, which establishes that $Alg^R(Cn_n) \subseteq HSP(M_n)$ and concludes the proof.

LEMMA 5. A REPRESENTATION THEOREM Each A in $HSP(M_n)$ is isomorphic with an algebra B in $SP(M_n)$.

Proof. Lemma 5 is an easy consequence of a result established by Grigolia /8/. Using C.C.Chang's MV-algebras as a starting point for his own considerations, he constructs, for each $n \geqslant 2$, the class of MV_n-algebras. It may be proved cf. /9/ that the class of MV_n-algebras coincides with that

of $Ł_n$-algebras, and then, by Lemma 4, with the class $HSP(M_n)$. In virtue of these identities, Lemma 5 is equivalent to the representation theorem proved by Grigolia for MV_n-algebras.

(Perhaps it is worth-while to notice that so-called Łukasiewicz algebras invented by Moisil fail to serve as $Ł_n$-algebras, at least to full extend, because, for $n > 4$, no operation definable in terms of the operations of those algebras can be regarded as an interpretation of the implication connective .)

§3. Some technical lemmas

Throughout this section the symbol A will always denote an algebra in $SP(M_n)$. Observe that elements of each such algebra can be defined to be functions of the form

$$a: T_A \longrightarrow M_n \,, \tag{5}$$

$a(t)$, $t \in T_A$, being called the projection of a onto the axis t. Similarly, $A(t)$ will denote the projection of the whole algebra A onto t. If $a \in A$, the symbol $\bar{a}$ will denote the subalgebra of A generated by a. Notice that

$$\bar{a}(t) \subseteq A(t) \subseteq M_n \,. \tag{6}$$

Since for each function a of the form $a: T \longrightarrow M_n$, there is an algebra A such that $a \in A$, the symbol $\bar{a}$ is defined for all such functions. In particular, if $m \in M_n$, $\bar{m}$ is the subalgebra of M_n generated by m.

LEMMA 6. For each A in $SP(M_n)$, and for each a in A, the algebra $\bar{a}$ is finite.

Proof. The set of all projections $a(t)$ is a subset of M_n and thus it is finite. Assume that $m_1,\ldots,m_k$ are all elements of it. The algebras $\bar{a}$ and $(\overline{m_1,\ldots,m_k})$ are isomorphic, and since the latter algebra is finite, $\bar{a}$ is also finite.

Assume that M_k is a subalgebra of M_n, 1 and m are elements of M_n, and $a \in A$ Put.

(7) $$\chi(M_k) = 1/k-1$$

(8) $$m^* = \chi(\bar{m})$$

(9) $$J(1,m) = \chi(M_{(1,m)}),$$

where $M_{(1,m)}$ is the subalgebra of M_n generated by the subset $\{1,m\}$.

With the help of McNaughton criterion /10/, one may verify that the operations $*$ and J are definable in the Łukasiewicz matrix M_n, and thus they are definable by the same identities (which, incidentely, are different for different n) in all algebras in $HSP(M_n)$, and in particular in all algebras in $SP(M_n)$. It makes sense then, given any a, $b \in A$, to write a^* or $J(a,b)$. Clearly $a^*(t) = (a(t))^* = \chi(\bar{a}(t))$ and $J(a,b)(t) = J(a(t),b(t))$.

Given any two elements a, b of an algebra A, define $a \simeq b$ to hold true whenever $\{\bar{a}(t): t \in T_A\} = \{\bar{b}(t): t \in T_A\}$. Clearly for each a, $a \simeq a^*$. We shall say that $a \in A$ is a <u>characteristic</u> element of A iff the following two conditions are satisfied:

(c1) $a = a^*$.

(c2) If for some b in A, $\bar{a}(t)$ is a subalgebra of $\bar{b}(t)$, for every $t \in T_A$, then $a \simeq b$.

LEMMA 7. For each algebra A in $SP(M_n)$ there is at least one element a characteristic for A.

Proof. Consider the quotient set $A/\simeq$. Clearly $A/\simeq$ is finite. Let $|a_1|,\ldots,|a_s|$ be all equivalence classes of $\simeq$. Select from the sequence $a_1,\ldots,a_s$ an element a such that for every a_i in the sequence, if $\bar{a}(t)$ is a subalgebra of $\bar{a}_i(t)$ for every t, then $a \simeq a_i$. Obviously such an element a exists. Now consider a^*. Since $a^* \simeq a$, and for each b in A,

there is a_i such that $b \simeq a_i$, we easily verify that a^* is a characteristic element of A.

LEMMA 8. If a is a characteristic element of A then $\bar{a} \sim A$.

Proof. $\bar{a}$ is a subalgebra of A, and thus we easily verify that $Cn_A \leqslant Cn_{\bar{a}}$. To prove the converse, suppose that for some formula α and for a set of formulas X,

$$\alpha \overline{\in} Cn_A(X) \ . \tag{10}$$

This assumption implies that there exists a avaluation h in A such that:

$$hX \subseteq \{1_A\} \ , \tag{11}$$

but

$$h\alpha \neq 1_A \ . \tag{12}$$

Let

$$\{A(t)\colon t \in T_A\} = \{A(t_1),\ldots,A(t_k)\} \ . \tag{13}$$

Put $h_t\alpha = (h\alpha)(t)$. Clearly for some t, $h_tX \subseteq \{1\}$ and $h_t\alpha \neq 1$. We may assume that this holds true for $t = t_1$. Observe now that for each $i = 1,\ldots,k$,

$$A(t_i) \in \{\bar{a}(t)\colon t \in T_A\} \ . \tag{14}$$

Indeed, select $b \in A$ such that $b(t_i) = \chi(A(t_i))$, and consider the element $J(a,b)$. Since $b(t_i) = \chi(A(t_i))$, then also $J(a,b) = \chi(A(t_i))$. On the other hand, from the definition of a characteristic element, and that of the operation J we almost immediately see that $a \simeq J(a,b)$. Thus the algebra generated by $\chi(A(t_i))$, i.e. the algebra $A(t_i)$ is in the set of projections of $\bar{a}$ on axes $t \in T_A$, which is exactly what (14) states.

In order to prove that $\alpha \overline{\in} Cn_{\bar{a}}(X)$ we have to define a valuation g in $\bar{a}$ such that $gX \subseteq \{1_{\bar{a}}\}$, and $g\alpha \neq 1_{\bar{a}}$. Observe that $1_A = 1_{\bar{a}}$. Since the language L is free in the class of all algebras similar to it and the set of propositional variables is the set of free generators of it in order to define g it is enough to define $g_t p$ for all $t \in T_A$, and for all propositional variables p in L.

Case 1. For all t such that $\bar{a}(t) = A(t_i)$ for some $i = 1,\ldots,k$, put

$$g_t p = h_{t_i} p, \tag{15}$$

for every variable p. Clearly for each such X, $g_t X \subseteq \{1\}$, but note that, in virtue of (15), for some t from those t's which are considered now $g_t \alpha = h_{t_1} \alpha \neq 1$.

Case 2. Consider an arbitrary t' such that $\bar{a}(t') \neq A(t_i)$ for any $i = 1,\ldots,k$. To establish the lemma we have to define g_t, in such a way that $g_{t'} X \subseteq \{1\}$. We shall show that, in the discussed case, for each variable p there exists t in T_A such that $h_t p \in \bar{a}(t')$. Thus, selecting any such t, we may put

$$g_{t'} p = h_t p \tag{16}$$

which clearly yields $g_{t'} X = h_t X \subseteq \{1\}$. Assuming that p is fixed, consider the element $b = J(a, hp)$. We easily see that $a \simeq b$, and hence $\bar{a}(t') = b(t)$, for some t, But, which one may easily verify, $h_t p \in \bar{b}(t)$, for each t. This implies that for some t, $h_t p \in \bar{a}(t')$ as it has been required in order to put (16).

The way in which g has been constructed guarantees that $g_t X \subseteq \{1\}$, for every X, but there is t such that $g_t \alpha \neq 1$. We have then: $g X \subseteq \{1_{\bar{a}}\}$ and $g\alpha \neq 1_{\bar{a}}$, which means that

$$\alpha \in Cn_{\bar{a}}(X). \tag{17}$$

Thus we have established that (10) implies (17) concluding the proof.

§4. The conclusion of the proof of the theorem

LEMMA 9. For each algebra A in $HSP(M_n)$ there are pairwise different submatrices $M_{m_1},\ldots,M_{m_k}$ of M_n such that $A \sim M_{m_1} \times M_{m_2} \times \ldots \times M_{m_k}$.

Proof. Each algebra A in $HSP(M_n)$ is isomorphic to an algebra B in $SP(M_n)$. Let a be a characteristic element of B. Then, $B \sim \bar{a}$, and hence $A \sim \bar{a}$. In turn $\bar{a}$ is isomorphic to the product $\bar{a}(t_1) \times \bar{a}(t_2) \times \ldots \times \bar{a}(t_k)$, where $a(t_1),\ldots,a(t_k) \in \{a(t): t \in T_B\}$ which conclude the proof.

Denote by $P_f(M_n)$ the set of all products of the form $M_{m_1} \times \ldots \times M_{m_k}$, where M_{m_i} are pairwise different submatrices of M_n. Clearly $P_f(M_n)$ is finite. Applying Lemma 9 we may pass from /III/ to

(IV) $$dm(Ł_n) \leqslant \{\text{card } \Theta: \Theta \subseteq P_f(M_n)/\sim\}$$

Since $P_f(M_n)$ is finite, (IV) yields

(V) $dm(Ł_n)$ is finite,

establishing the theorem of the paper.

REFERENCES

/1/ R. Wójcicki, ON MATRIX REPRESENTATIONS OF CONSEQUENCE OPERATIONS OF ŁUKASIEWICZ SENTENTIAL CALCULI, Zeitschrift fuer Mathematische Logik und Grundlagen der Mathematik, 19(1973), pp.239-247.

/2/ J. Łoś and R. Suszko, REMARKS ON SENTENTIAL LOGICS, Indagationes Mathematicae, 20(1962), pp.426-436.

/3/ R. Wójcicki, SOME REMARKS ON CONSEQUENCE OPERATION IN SENTENTIAL LOGICS, Fundamenta Mathematicae, LXVIII(1970) pp.268-279.

/4/ - , MATRIX APPROACH IN METHODOLOGY OF SENTENTIAL CALCULI, Studia Logica, 32(1973), pp.168-195.

/5/ - , THE LOGICS STRONGER THAN THE THREE VALUED ŁUKASIEWICZ SENTENTIAL CALCULUS, Studia Logica, 33(1974), pp.201-214.

/6/ G. Malinowski, DEGREAS OF MAXIMALITY OF SOME ŁUKASIEWICZ LOGICS, Bulletin of the Section of Logic, Polish Academy of Sciences, 3(1974)

/7/ H. Rasiowa, AN ALGEBRAIC APPROACH TO NON CLASSICAL LOGICS, North Holland Publ. Co., 1974

/8/ R.S. Grigolia, AN ALGEBRAIC ANALYSIS OF n-VALUED LOGICAL SYSTEMS OF ŁUKASIEWICZ AND TARSKI /in Russian/, Proceedings of Tbilisi University, A 6-7(1973) pp.121-132

/9/ G. Malinowski, S-ALGEBRAS FOR n-VALUED SENTENTIAL CALCULI OF ŁUKASIEWICZ, Bulletin of the Section of Logic, Polish Academy of Sciences, 3(1974)

/10/ R. McNaughton, A THEOREM ABOUT INFINITE-VALUED SENTENTIAL LOGIC, Journal of Symbolic Logic, 16(1951), pp.1-13.

MATRIX REPRESENTATION FOR THE DUAL COUNTERPARTS OF ŁUKASIEWICZ n-VALUED SENTENTIAL CALCULI AND THE PROBLEM OF THEIR DEGREES OF MAXIMALITY

Grzegorz Malinowski
Łódź University
Poland

Introduction

0.1. Dual counterparts of the consequence operations. Let us consider any sentential language (algebra of formulas) $\underline{L}_S = (L_S, F_1, \ldots, F_n)$ and a structural consequence operation C defined over L_S (cf. [12]). Put $S = (\underline{L}_S, C)$. The pair S will be called a *sentential calculus*. An element $\alpha \in C(\emptyset)$ will be called a *theorem of* S. By the *consequence dual with respect to* C we shall understand, following R. Wójcicki [10], the consequence defined as follows:

(1) $\alpha \in dC(X)$ if and only if there exists a finite set $Y \subseteq X$ such that

$$\bigcap \{C(\{\beta\}) : \beta \in Y\} \subseteq C(\{\alpha\})$$ [1]

Observe that (1) can be treated as a definition of a function d defined on the set of all consequences (of a given language). If $S = (\underline{L}_S, C)$ is a sentential calculus, then $dS = (\underline{L}_S, dC)$ will be called a *calculus dual with respect to* S.

The concept of the dual consequence refers to certain ideas of Łukasiewicz who investigated (cf. [3]) some rules leading from falsity to falsity. Indeed, under certain assumptions the operation dC may be treated as one permitting derivation of false sentences from other false sentences.

0.2. Dual counterparts of Łukasiewicz sentential calculi. Let L be the set of all formulas built up in the usual way by means of sentential variables $p,q,r,\ldots$ and connectives $\rightarrow, \vee, \wedge, \neg$. The algebra of formulas

(2) $$\underline{L} = (L, \rightarrow, \vee, \wedge, \neg)$$

will be called the *language of Łukasiewicz sentential calculi*. Let us consider the algebras

(3) $$\underline{A}_n = (A_n, \rightarrow, \vee, \wedge, \neg)$$

(where n is an arbitrary natural number ≥ 2) similar to $\underline{L}$ and defined as follows:

(i) $A_n = \{0, \frac{1}{n-1}, \ldots, \frac{n-2}{n-1}, 1\}$

(ii) $x \rightarrow y = \min(1, 1-x+y)$; $x \vee y = \max(x,y)$; $x \wedge y = \min(x,y)$; $\neg x = 1-x$.

[1] As a matter of fact the consequence dC coincides with $d_\omega C$ defined in [10].

Given n,

(4) $$M_n = \langle \underline{A}_n , \{1\} \rangle$$

is the well known *n-valued Łukasiewicz matrix* (cf. [4]). Every homomorphism $h : \underline{L} \to \underline{A}_n$ will be called a *valuation of formulas* of the language $\underline{L}$ in the matrix M_n. For any $X \subseteq L$, let us put

(5) $$C_n(X) = \{\alpha \in L : \text{for every } h : \underline{L} \to \underline{A}_n, \text{ if } h(X) \subseteq \{1\}, \text{ then } h\alpha = 1\}.$$

C_n will be called a *matrix consequence of* M_n. The pair $t_n = (\underline{L} , C_n)$ will be called the *n-valued Łukasiewicz calculus*. Let us put

(6) $$\overline{M}_n = \langle \underline{A}_n , A_n - \{1\} \rangle.$$

Also, for any $X \subseteq L$,

(7) $$\overline{C}_n(X) = \{\alpha \in L : \text{for every } h : \underline{L} \to \underline{A}_n, \text{ if } h(X) \subseteq A_n - \{1\}, \text{ then } h\alpha \neq 1\}.$$

Then there holds (cf. [7])

THEOREM 1. $dC_n = \overline{C}_n$ $(n \geq 2)$.

From the last theorem it follows that $dt_n = (\underline{L}, \overline{C}_n)$. In the sequel if we deal with dt_n we will write dC_n, but instead of the consequence dC_n we will always use its matrix characterization $\overline{C}_n$ as more convenient.
Now, we shall give a brief account of some special connectives of $\underline{L}$ that are very important for our further investigations. Let $p \in L$ be an arbitrary sentential variable. Let us put

(8) $$\neg_n p = \begin{cases} \neg p & \text{for } n=2 \\ p \xrightarrow[n-2]{} (p \wedge \neg p) & \text{for } n \geq 3 , \end{cases}$$

where $\alpha \xrightarrow[0]{} \beta =_{df} \beta$ and $\alpha \xrightarrow[n]{} \beta =_{df} \alpha \to (\alpha \xrightarrow[n-1]{} \beta)$.
Let us notice that the function $\neg_n$ in $\underline{A}_n$ corresponding to the above connective is the following:

(9) $$\neg_n x = \begin{cases} 0 & \text{if } x=1 \\ 1 & \text{otherwise} . \end{cases}$$

As a rule, we will always write $\neg$ instead of $\neg_n$ while the given n-valued calculus is considered. The following correspondence between the sets of the theorems of t_n and dt_n will be used often in the paper:

(10) $\neg\alpha \in dC_n(\emptyset)$ if and only if $\alpha \in C_n(\emptyset)$.

0.3. The problem of the paper. Let $S = (\underline{L}_S, C)$ be a sentential calculus. Any couple $M = \langle \underline{A}_M, I_M \rangle$, where $\underline{A}_M$ is an algebra similar to the language $\underline{L}_S$ and I_M is a subset of the set of elements A_M of $\underline{A}_M$, will be called a *matrix corresponding to* $\underline{L}_S$. For any $X \subseteq L_S$, let us put

(11) $$Cn_M(X) = \{\alpha \in L_s : \text{for every homomorphism } h : \underline{L}_s \longrightarrow \underline{A}_M \text{ , if } h(X) \subseteq I_M \text{ , then } h\alpha \in I_M\}.$$

Then, Cn_M is a *consequence determined by the matrix* M *in the language* $\underline{L}_s$. M will be called an S-*matrix* provided that $C(X) \subseteq Cn_M(X)$ for every $X \subseteq L_s$ or equivalently $C \leq Cn_M$. The class of all S-matrices for a given calculus $S = (\underline{L}_s, C)$ will be denoted by Matr(C). An S-matrix M will be called an S-*algebra* provided that (cf. [12]) I_M is a one-element set. Dealing with implicative calculi in the sense of Rasiowa (cf. [8]) it is convenient to consider a subclass $Alg^R(C)$ (of Matr(C)) consisting of those S-algebras which satisfy the following condition: for any arbitrary elements a and b, if $a \rightarrow b = 1$ and $b \rightarrow a = 1$, then $a = b$ (where 1 is the distinguished element). The subclass $Alg^R(C)$ described above is exactly the class of S-*algebras of Rasiowa* (cf. [8]).

The aim of this paper is to give a characterization of the class $Matr(dC_n)$ and, based on this work, to give a characterization of the strengthenings of the calculi dt_n, consequently establishing their degrees of maximality, cf. [11]. Although the calculi dt_n are not implicative, we shall use in our considerations matrices analogous to the S-algebras of Rasiowa. The class of all such matrices algebras, namely $Matr^R(dC_n)$ is constructed and studied.

PART I. *MATRICES*

I.1. Some results concerning $Alg^R(C_n)$. Let $K(\underline{A}_n)$ (given n, $n \geq 2$) denote the smallest equational class containing the algebra $\underline{A}_n$ (see (3)). Using Birkhoff's characterization of equational classes we have

(12) $$K(\underline{A}_n) = H(S(P(\underline{A}_n))) \ ,$$

where P denotes the operation of taking direct products, S of subalgebras and H of homomorphic images. Let us also denote by $K(M_n)$ the same class of algebras but treated as matrices with one distinguished element 1. This element is a natural counterpart of $1 \in A_n$.

THEOREM 2. (cf. [13]). $K(M_n) = Alg^R(C_n)$.

The axiomatization of $Alg^R(C_n)$ was given by Grigolia in [2] (for that matter see [6]). The following theorem is another formulation of the representation theorem given by Grigolia:

THEOREM 3. (cf. [2]). Every algebra of the class $K(M_n)$ is isomorphic with some subalgebra of a direct product M_n^T (T is a set of indices).

I.2. Construction of the class $Matr^R(dC_n)$. Let $M \in Matr(dC_n)$. This means that $M = \langle \underline{A}_M, I_M \rangle$, where $\underline{A}_M$ is an algebra similar to the language $\underline{L}$, I_M is some subset of A_M, and moreover (see 0.3), $dC_n \leq Cn_M$.

DEFINITION 1. If $M \in Matr(dC_n)$, then we put

(13) $a \approx_M b$ if and only if $\neg(a \to b), \neg(b \to a) \in I_M$.

LEMMA 1. For $M \in \mathrm{Matr}(dC_n)$, $\approx_M$ is a (matrix) congruence of M.[1]

PROOF. For every sentential variable p, $\neg(p \to p) \in dC_n(\emptyset)$, so $\neg(a \to a) \in I_M$ for any $a \in A_M$, and $a \approx_M a$. The symmetry of the condition (13) implies the symmetry of $\approx_M$. Let us now assume that for some $a,b,c \in A_M$ $a \approx_M b$ and $b \approx_M c$. By the definition, it means that

$$\neg(a \to b), \neg(b \to a) \in I_M ,$$
$$\neg(b \to c), \neg(c \to b) \in I_M .$$

This is easy to see that

$$\frac{\neg(p \to q)\quad \neg(q \to r)}{\neg(p \to r)}$$

is a rule of dC_n and thus a rule of Cn_M. So, from our assumptions, we obtain $\neg(a \to c)$, $\neg(c \to a) \in I_M$, which means that $a \approx_M c$. This ends the proof of the fact that $\approx_M$ is an equivalence relation on A_M. To complete the proof that $\approx_M$ is in fact a matrix congruence, it must be shown that: (a) $a \to c \approx_M b \to d$, $a \vee c \approx_M b \vee d$, $a \wedge c \approx_M b \wedge d$ and $\neg a \approx_M \neg b$, whenever $a \approx_M b$ and $c \approx_M d$; (b) If $a \approx_M b$, then either both a and b are in I_M or neither of them is in I_M. The simple proof of (a) is based on the following rules of dC_n and thus of Cn_M :

$$\frac{\neg(p \to q)}{\neg(\neg q \to \neg p)}, \qquad \frac{\neg(p \to q), \neg(q \to p), \neg(r \to s), \neg(s \to r)}{\neg[(p \to r) \to (q \to s)], \neg[(q \to s) \to (p \to r)]},$$

$$\frac{\neg(p \to q)\quad \neg(r \to s)}{\neg(p \vee r \to q \vee s)}, \qquad \frac{\neg(p \to q)\quad \neg(r \to s)}{\neg(p \wedge r \to q \wedge s)} .$$

Now, we will prove that (b) holds for $\approx_M$. Let us assume that $a \in I_M$, $b \in I_M$ and $a \approx_M b$. According to the fact that

$$\frac{\neg(p \to q)\quad q}{p}$$

is a rule of dC_n, we obtain $b \in I_M$. This contradics the previous assumption and thus ends the proof.

Now, we will define a special subset of elements of the matrix M.

DEFINITION 2. $V_M = \{a \in A_M$: there exists a formula $\alpha \in L$, such that $\neg\alpha \in dC_n(\emptyset)$, and such that there exists a homomorphism $h: L \to M$, for which $h\alpha = a\}$.

LEMMA 2. If $M \in \mathrm{Matr}(dC_n)$, then the quotient $V_M/\approx_M = \{|a| : a \in V_M\}$ is a one-element set and it will be denoted by 1_M.

PROOF. Let $a,b \in V_M$. According to Definition 2 there exist formulas $\alpha,\beta \in L$

[1] For the notion of a matrix congruence see [12].

and there exist homomorphisms $h_\alpha : \underline{L} \to M$, $h_\beta : \underline{L} \to M$ such that $\neg\alpha, \neg\beta \in dC_n(\emptyset)$ and $h_\alpha(\alpha) = a$, $h_\beta(\beta) = b$. Let $VAR(\alpha) = \{p_1,\ldots,p_n\}$, $VAR(\beta) = \{q_1,\ldots,q_m\}$ denote the sets of all variables appearing in α and β, respectively. Now let us exchange the variables of the set $VAR(\beta)$ putting q_i / q_i^* so that $VAR(\alpha) \cap \{q_1^*,\ldots,q_m^*\} = \emptyset$. It is obvious that such a construction is possible. Let β^* be a formula resulting from β under the above exchange. Note that $\neg\beta^* \in dC_n(\emptyset)$. Moreover, $\neg(\alpha \to \beta^*)$, $\neg(\beta^* \to \alpha) \in dC_n(\emptyset)$. Now, consider a mapping $h : VAR(L) \to A_M$ such that $h(p_i) = h_\alpha(p_i)$, $h(q_i^*) = h_\beta(q_i)$.

This mapping induces a homomorphism $H : \underline{L} \to \underline{A}_M$, and

$$H(\alpha \to \beta^*) = H(\alpha) \to H(\beta^*) = a \to b ,$$
$$H(\beta^* \to \alpha) = H(\beta^*) \to H(\alpha) = b \to a .$$

Due to the previous part of the proof we have $\neg(a \to b) \in I_M$ and $\neg(b \to a) \in I_M$, and thus $a \approx_M b$.

LEMMA 3. Let $M \in Matr(dC_n)$ and let $\approx_M$ be the relation given by (13). Then for any $|a|, |b| \in {}^M/_{\approx_M}$ the following condition holds:

(14) If $|a| \to |b| = \underline{1}_M$ and $|b| \to |a| = \underline{1}_M$, then $|a| = |b|$.

PROOF. Let us assume that $|a| \to |b| = |b| \to |a| = \underline{1}_M$. Then $|a \to b| = |b \to a| = \underline{1}$, and therefore

(*)
$$\neg[(a \to b) \to k] \quad , \quad \neg[k \to (a \to b)] \quad \in I_M$$
$$\neg[(b \to a) \to k] \quad , \quad \neg[k \to (b \to a)] \quad \in I_M$$

for any $k \in V_M$. Let us take for example any element k which corresponds to the formula $p \to p$ ($\neg(p \to p) \in dC_n(\emptyset)$ implies that such k exists). The rule

$$\frac{\neg[(r \to s) \to (p \to p)] \quad \neg[(p \to p) \to (r \to s)]}{\neg(r \to s)}$$

is a rule of dC_n , and therefore of Cn_M. Thus, from (*), we obtain $\neg(a \to b) \in I_M$ and $\neg(b \to a) \in I_M$, which means that $a \approx_M b$. Therefore $|a| = |b|$.

As we have noticed in 0.2, the following equivalence holds: $\neg\alpha \in dC_n(\emptyset)$ if and only if $\alpha \in C_n(\emptyset)$. Consequently, any homomorphic image (a valuation) of an arbitrary theorem of the n-valued sentential calculus of Łukasiewicz in every matrix ${}^M/_{\approx_M}$, where $M \in Matr(dC_n)$, is $\underline{1}_M$. This point justifies the acceptance of our next definition.

DEFINITION 3. $Matr^R(dC_n) = \{{}^M/_{\approx_M} : M \in Matr^R(dC_n)\}$.

The class $Matr^R(dC_n)$ has similar properties to those of the class $Alg^R(C_n)$. In particular, as it will be shown, so-called Lindenbaum matrix for dC_n is free in this class.

1.3. The equational characterization of $Matr^R(dC_n)$. Let $K(\underline{A}_n)$ be the equational class defined in 1.1. Now, let us denote by $K(\overline{M}_n)$ a corresponding class of matrices defined in the following manner. For every $\underline{A}_M \in K(\underline{A}_n)$ we take the matrix $M = \langle \underline{A}_M, I_M \rangle$, where I_M is determined by one of the conditions:

(i_1) If $\underline{A}_M = \underline{A}_n$, then $I_M = A_n - \{1\}$,

(i_2) If $\underline{A}_M$ is a product of the indexed set of algebras $\{\underline{A}_{M_i}\}_{i \in T}$, then $I_M = \Pi \{I_{M_i} : i \in T\}$,

(i_3) If $\underline{A}_M$ is a subalgebra of some algebra $\underline{A}_{M_o}$, then $I_M = \underline{A}_M \cap I_{M_o}$,

(i_4) If $\underline{A}_M = h(\underline{A}_{M_o})$, where h is a homomorphism, then $I_M = h(I_{M_o})$.

In the sequel, the element of $\underline{A}_M$ being a natural counterpart of $1 \in A_n$ will be denoted by 1_M.

LEMMA 4. If $M = \langle \underline{A}_M, I_M \rangle \in K(\overline{M}_n)$, then $I_M = \{a \in A_M : \neg a = 1_M\}$.

PROOF. This lemma follows simply from definition of $K(\overline{M}_n)$ and the properties of $\neg$ (see (9)).

LEMMA 5. For every matrix $M = \langle \underline{A}_M, I_M \rangle \in K(\overline{M}_n)$ there exists a matrix $M^\bullet = \langle \underline{A}_{M^\bullet}, I_{M^\bullet} \rangle$ with $\underline{A}_{M^\bullet}$ being a subalgebra of $\underline{A}_n^T$ (T is some set of indices), such that $\underline{A}_M$ is isomorphic with $\underline{A}_{M^\bullet}$ and $Cn_M = Cn_{M^\bullet}$.

PROOF. First we note that if a matrix $M = \langle \underline{A}_M, I_M \rangle$ is such that $\underline{A}_M$ is a subalgebra of the product $\underline{A}_n^T$ then for every $a \in A_M$ there holds

(**) $\quad a \in I_M$ if and only if $\neg a = 1_M$.

According to Theorem 3, an arbitrary $\underline{A}_M$ is isomorphic with some subalgebra of some product $\underline{A}_n^T$. Let $\underline{A}_M \simeq_f \underline{A}_{M^\bullet} \subseteq \underline{A}_n^T$ (i. e. f establishes the isomorphism of $\underline{A}_M$ and $A_{M^\bullet}$). Then to prove our lemma, it suffices to prove that $f(I_M) = I_{M^\bullet}$ (cf. [12]).
Let us assume now that $x \in I_M$. Then $\neg x = 1_M$ (Lemma 4) and $f(\neg x) = 1_{M^\bullet}$ (because $f(1_M) = 1_{M^\bullet}$). Therefore $\neg f(x) = 1_M$ and from (**) we obtain $f(x) \in I_{M^\bullet}$. So $f(I_M) = I_{M^\bullet}$. This means that $Cn_M = Cn_{M^\bullet}$.

From Lemma 4 and from the part of the proof of Lemma 5 concerning the isomorphism f we easily obtain

THEOREM 4. $M = \langle \underline{A}_M, I_M \rangle \in K(\overline{M}_n)$ if and only if 1° $\underline{A}_M \in K(\underline{A}_n)$ and 2° $I_M = \{a \in \underline{A}_M : \neg a = 1_M\}$.

LEMMA 6. $K(\overline{M}_n) \subseteq Matr^R(dC_n)$.

PROOF. The class $Matr(dC_n)$ is closed under the operations of forming submatrices and direct products (cf. [12]). Therefore, by Lemma 5, we obtain $K(\overline{M}_n) \subseteq Matr(dC_n)$. Now, let us consider an arbitrary $M \in K(\overline{M}_n)$ and the relation $\approx_M$ defined in the usual way. Let us assume that for some a,b A_M, $a \approx_M b$. Thus $\neg(a \rightarrow b), \neg(b \rightarrow a) \in I_M$.
In any matrix that is a submatrix of a product of matrices $\overline{M}_n$ the following condition holds: If $x \rightarrow y = y \rightarrow x = 1_M$, then $x = y$. Therefore, by properties of the isomorphism appearing in the proof of Lemma 5 and by (**), we obtain $a \rightarrow b = b \rightarrow a = 1_M$ and consequently $a = b$. What we have proved is $M / \approx_M$. So $M \in Matr^R(dC_n)$. This concludes the proof of our lemma.

DEFINITION 4. In the language $\mathcal{L}$ we introduce the following relation:

(15) $\quad \alpha \approx \beta$ if and only if $\neg(\alpha \rightarrow \beta), \neg(\beta \rightarrow \alpha) \in dC_n(\emptyset)$.

One can easily verify that $\approx$ is a congruence relation on $\mathcal{L}$.

The quotient matrix

(16) $$\Lambda_n = \langle \underline{L}/_{\approx}, dC_n(\emptyset)/_{\approx} \rangle$$

will be called the *Lindenbaum matrix for* dt_n. The one-element set $1_\Lambda = C_n(\emptyset)/_{\approx}$ is the greatest element of the algebra $\underline{L}/_{\approx}$ with respect to the order given as follows:

(17) $$|\alpha| \leq |\beta| \quad \text{if and only if} \quad \neg(\alpha \rightarrow \beta) \in dC_n(\emptyset) .$$

As a matter of fact the relation $\approx$ defined by (15) coincides with the relation $\dot{\approx}$ defined on $\underline{L}$ in the following manner: $\alpha \dot{\approx} \beta$ if and only if $(\alpha \rightarrow \beta)$, $(\beta \rightarrow \alpha) \in C_n(\emptyset)$. And the order $\leq$ given by (17) coincides with the order $\dot{\leq}$ defined as follows : $|\alpha| \dot{\leq} |\beta|$ if and only if $(\alpha \rightarrow \beta) \in C_n(\emptyset)$ - for these matters see [10] . Taking the above into account we have (cf. [13])that the Lindenbaum matrix $\langle \underline{L}/_{\approx}, C_n\emptyset/_{\approx} \rangle$ belongs to the class $K(M_n)$. Therefore $\underline{L}/_{\approx} \in K(\underline{A}_n)$.

LEMMA 7. $\Lambda_n \in K(\overline{M}_n)$.

PROOF. From the equivalence $\alpha \in dC_n(\emptyset)$ if and only if $\neg\alpha \in C_n(\emptyset)$, it follows that $|\alpha| \in dC_n(\emptyset)/_{\approx}$ if and only if $\neg|\alpha| = |\neg\alpha| = 1_\Lambda$. Thus

(18) $$dC_n(\emptyset)/_{\approx} = \{ |\alpha| : \neg|\alpha| = 1_\Lambda \} .$$

According to Theorem 4, $\Lambda_n \in K(\overline{M}_n)$.

H. Rasiowa in [8] proved that the Lindenbaum matrix for any consistent implicative calculus S is free in the class $Alg^R(S)$. This result is also valid for Łukasiewicz calculi. In view of the remarks given above (before Lemma 7), we see that the following lemma holds:

LEMMA 8. Λ_n is free in the class $Matr^R(dC_n)$. The free generators of Λ_n are the classes determined by the sentential variables.

The algebra which belongs to the equational class X and is free (in X)generates the whole class X (cf. [1]). Thus lemmas 7,8 give us the following

LEMMA 9. $Matr^R(dC_n) \subseteq K(\overline{M}_n)$.

Combining the results of Lemma 6 and Lemma 9 we obtain the main theorem of the present section.

THEOREM 5. $Matr^R(dC_n) = K(\overline{M}_n)$

PART II. *STRENGTHENINGS*

II.1. Degrees of maximality of Łukasiewicz n-valued calculi. If $S = (\underline{L}_s, C)$ is a sentential calculus, then the calculus $S' = (\underline{L}_s, C')$ will be called a *strengthening of* S provided that $C \leq C'$. The cardinal number of the set of all strengthenings of a given calculus S will be called the *degree of maximality of* S.

R. Wójcicki established in [11] that the degree of maximality of the three - valued calculus of Łukasiewicz equals 4. The author of the present paper gave in [5] a different proof of this fact and moreover, using a new method, established that the degree of maximality of any n-valued calculus of Łukasiewicz such that n-1 is prime

is also equal to 4. Finally, R. Wójcicki showed in [13] that the degree of maximality of any n-valued calculus of Łukasiewicz is finite. The aim of our next section is to give an analogous result for the dual counterparts of Łukasiewicz calculi. The method used here will not differ essentially from that used by Wójcicki. Now we will present the slightly modified version of the theorem that was a key to the general solution of the problem of the degrees of maximality of Łukasiewicz calculi. This theorem will be used in our later considerations.

Given a Łukasiewicz matrix M_n, let us consider the set ∇_n built up in the following way:

1° If M_k is a submatrix of M_n, then $M_k \in \nabla_n$,

2° Let $M_{k_i} = \langle \underline{A}_{k_i}, \{1\} \rangle$ as usual. If $M_{k_1}, \ldots, M_{k_s}$ is a set of distinct submatrices of M_n, then $\langle \Pi\{\underline{A}_{k_i} : i=1,\ldots,s\}, \{1_\Pi\} \rangle \in \nabla_n$,

3° The set ∇_n contains no other matrices but those described in 1° and 2°.

Clearly, $\nabla_n \subsetneq K(M_n)$.

LEMMA 10. (cf. [13]). For every $M \in K(M_n)$ there exists a matrix $M^* \in \nabla_n$ such that $Cn_M = Cn_{M^*}$.

Let us remark that the above lemma implies that the matrices of $K(M_n)$ are able to determine only *a finite number of distinct consequences.*

II.2. Degrees of maximality of $d\mathcal{L}_n$. Let $M \in Matr(dC_n)$ be an arbitrary matrix and let $\approx_M$ be the congruence relation defined by (13). From the fact that $\approx_M$ is a congruence of M (Lemma 1) it follows (cf. [12]) that

$$Cn_M = Cn_{M/\approx_M}. \tag{19}$$

As it was defined in the previous part of the paper, the whole class of such $M/\approx_M$ is $Matr^R(dC_n)$. Moreover, according to the equational characterization given in I.3 we have $Matr^R(dC_n) = K(\overline{M}_n)$. This implies that the consequence operation determined by an arbitrary matrix $M \in Matr(dC_n)$ is equal to some consequence determined by the matrix belonging to the class $K(\overline{M}_n)$.

LEMMA 11. Let $M = \langle \underline{A}_M, 1_M \rangle \in K(\overline{M}_n)$ and let $M_+ = \langle \underline{A}_M, 1_M \rangle$ be the corresponding matrix from $K(M_n)$. Then

$$\begin{aligned} &\alpha \in Cn_M(X) \quad \text{if and only if} \quad \neg\alpha \in Cn_{M_+}(\neg X), \\ &\neg\alpha \in Cn_M(\neg X) \quad \text{if and only if} \quad \alpha \in Cn_{M_+}(X) \end{aligned} \tag{20}$$

where $X \subseteq L$ and $\neg X$ denotes the set resulting from X by preceding each of its formulas by $\neg$.

PROOF. This lemma is a simple consequence of the characterization of 1_M that has been given in Theorem 4 (I.3).

Now let us consider the set $\overline{\nabla}_n$ (of matrices) being the natural counterpart of ∇_n which was constructed in II.1. If $M \in \nabla_n$ and $M = \langle A_M, 1_M \rangle$, then the corresponding $M_- \in \overline{\nabla}_n$ is of the form $M_- = \langle A_M, I_M \rangle$, where $I_{M_-} = \{ a \in A_M : \neg a = 1_M \}$.

LEMMA 12. For every $M \in K(\overline{M}_n)$ there exists a matrix $M^* \in \overline{\nabla}_n$ such that $Cn_M = Cn_{M^*}$.

PROOF. According to Lemma 11 we have the following conditions equivalent in sequence

(i_1) $\alpha \in Cn_M(X)$, (i_2) $\neg\alpha \in Cn_{M_+}(\neg X)$, (i_3) $\neg\alpha \in Cn_{M^*_+}(\neg X)$, (i_4) $\alpha \in Cn_{M^*}(X)$,

(M_+ and M^*_+ are taken as in Lemma 10 and Lemma 11); $(i_1) \Longleftrightarrow (i_4)$ concludes the proof.

From the last lemma and from the remarks given at the beginning of this section we obtain

COROLLARY 1. For every $M \in Matr(dC_n)$ there exists a matrix $M_\delta \in \overline{\nabla}_n$ such that $Cn_M = Cn_{M_\delta}$. Becouse of the fact that the set $\overline{\nabla}_n$ is finite we have that there exists only a finite number of distinct consequences determined by the matrices of the class $Matr(dC_n)$.

THEOREM 6. The degree of maximality of any (finite) n-valued logic dt_n is finite.

PROOF. A sentential calculus $(\underline{L}, C)$ is a strengthening of the calculus dt_n provided that $C \geq dC_n$. According to the definition of a sentential calculus, C must be a structural consequence on L. Therefore, there exists (see e. g. [12], p. 20) the class of matrices $N = \{N_t\}_{t \in T}$ (T is a set of indices) such that $C = Cn_N$, i. e. for every $X \subseteq L$

$$C(X) = \bigcap_{t \in T} Cn_{N_t}(X) .$$

Then we see that for every $t \in T$ $dC_n \leq Cn_{N_t}$. Therefore the class N is equivalent (in the sense of determining the same consequence) to some subclass of $Matr(dC_n)$. Corollary 1 implies that the classes N which determine the distinct consequences on L must be finite in number. So, the number of strengthenings of dt_n is finite. In other words - the degree of maximality of any calculus dt_n is finite.

A moment of reflection over Lemma 11 related to the sets ∇_n and $\overline{\nabla}_n$ leads us to the following

COROLLARY 2. The numbers of strengthenings of the calculi t_n and dt_n are equal. In particular, for the calculi t_n, dt_n for which $n-1$ is prime, the degree of maximality equals 4(cf. [5]).

Final Remark. As a matter of fact the calculi dt_n may be translated into some n-valued calculi from Rosser and Turquette's collection (cf. [9]). Then the following problem arises: How to find similar characterization of other logics from [9] and to describe their strengthenings.
An open question: Is it possible to give a characterization of every Rosser and Turquette logic in terms of Matr(C) and strengthenings?

REFERENCES

[1] P. M. Cohn, *Universal algebra*, Harper and Roe, New York, 1965.

[2] R. S. Grigolia, *Algebraic analysis of Łukasiewicz-Tarski's n-valued logical systems in Russian* , Proceedings of Tbilisi University, A 6-7 149-150 , Mathematical and Natural Sciences, Tbilisi (1973), pp. 121-132.

[3] J. Łukasiewicz, *Aristotle's syllogistic from the standpoint of modern formal logic*, Oxford, 1951.

[4] J. Łukasiewicz, A. Tarski, *Untersuchungen über den Aussagenkalkül*, Comptes--rendus des séances de la Société de Sciences et de Lettres de Varsovie, Cl.III, 23(1930), pp. 30-50.

[5] G. Malinowski, *Degrees of maximality of some Łukasiewicz logics*, Bulletin of The Section of Logic, Institute of Philosophy and Sociology Polish Academy of Sciences, vol. 3(1974), no. 3/4, pp. 27-33.

[6] G. Malinowski, *S-algebras for n-valued sentential calculi of Łukasiewicz*, Bulletin of The Section of Logic, Institute of Philosophy and Sociology Polish Academy of Sciences, vol. 3(1974), no. 2, pp. 25-30.

[7] G. Malinowski, M. Spasowski, *Dual counterparts of Łukasiewicz's sentential calculi*, Studia Logica, XXXIII(1974), no. 2, pp. 153-162.

[8] H. Rasiowa, *An algebraic approach to non-classical logic*, North Holland Publishing Company, Amsterdam, Polish Scientific Publishers, Warszawa, 1974.

[9] J. B. Rosser, A. R. Turquette, *Many-valued logics*, North Holland Publishing Company, Amsterdam, 1952.

[10] R. Wójcicki, *Dual counterparts of consequence operations*, Bulletin of The Section of Logic, Institute of Philosophy and Sociology Polish Academy of Sciences, vol. 2(1973), pp. 54-57.

[11] R. Wójcicki, *The logics stronger than three valued sentential calculus. The notion of degree of maximality versus the notion of degree of completeness*, Studia Logica, XXXIII(1974), no. 2, pp. 201-214.

[12] R. Wójcicki, *Matrix approach in methodology of sentential calculi*, Studia Logica, XXXII(1973), pp. 7-37.

[13] R. Wójcicki, *A theorem on the finiteness of the degree of maximality of the n-valued Łukasiewicz logics*, this volume, pp.

SOME APPLICATIONS OF A GENERAL THEORY OF DIGRAPH MEASURES

John C. Hansen
University of Missouri-Rolla
U.S.A.

1. INTRODUCTION

It is a simple matter to construct digraphs with clearly defined properties, such as transitivity, symmetry, and balance. It is less simple to construct digraphs which are accurate representations of complex processes in the real world. This is because such processes oftentimes possess properties of the above mentioned kind in ways that all ill-defined and less than complete. (Something especially true in the behavioral sciences, where applications of graph theory are very popular.)

A social situation normally characterized by a signed digraph is rarely fully balanced. A preference schedule in a comparison test is seldom completely transitive. Models which ignore such exigencies are not realistic.

Graph-theoretic arguments cannot be brought to bear on substantive problems in substantive ways if they are stated within the context of graphtheoretic models which are, themselves, manifestly unrealistic. Thus, the rationale is clear for seeking to formulate graph theories that admit orders or measures of degrees of transitivity, symmetry, and the like.

This idea is not totally new. The attempt of Harary, Norman, and Cartwright [2] to develop a measure of the degree of balance in signed digraphs is widely recognized as pioneering; and the work of Norman and Roberts [3] is clearly an important elaboration of this effect.

Until now, however, no systematic attempt has been made to generalize the work of Norman et al. to include methods for constructing measures for graph-theoretic properties other than balance.

This is accomplished in Part Two of the present paper; in Part Three specific measures for transitivity and symmetry, in addition to balance, are obtained, and illustrated by example; and in Part Four the correspondence between these measures, fuzzy sets and n-valued logic is pointed out.

2. A GENERAL THEORY OF DIGRAPH MEASURES

2.1 A Discussion of L and Norms on L

Definition 2.1.2 Let L be the class of all infinite sequences of non-negative integers with only finitely many nonzero terms. Members of L will be denoted by capital letters from the beginning of the alphabet, with terms in the sequence being indicated by properly subscripted lower case letters (e.g. $A = (a_1, a_2, \ldots)$).

Definition 2.1.2 Let N be the class of natural numbers.

Definition 2.1.3 Let I be the class of non-negative integers.

Definition 2.1.4 Let R be the class of positive real numbers.

It is possible to define a multiplication of sequences in L by integers in I.

Definition 2.1.5 Let $A \in L$ and $X \in I$; then $xA = (xa_1, xa_2, \ldots)$.

Definition 2.1.6 Let $A, B \in L$; then $A + B = (a_1, a_2, \ldots) + (b_1, b_2, \ldots) = (a_1+b_1, a_2+b_2, \ldots)$.

If iA is thought of as a type of scalar multiplication, the system considered here is almost a module. If fails with regard to additive inverses in L and additive inverses in I. It is still possible to write $A-B = (a_1-b_1, a_2-b_2, \ldots)$, but there is no guarantee of its existence.

Definition 2.1.7 A norm on L is a real-valued function d satisfying the following properties for all $x \in I$ and $A, B \in L$.

1) $d(A) = 0$ if $A = 0$, $d(A) > 0$ if $A \neq 0$.
2) $d(xA) = xd(A)$.
3) $d(A + B) \leq d(A) + d(B)$.

Definition 2.1.8 I_n is the sequence whose terms $i_m = 0$ if $n \neq m$ and $i_m = 1$ if $n = m$.

Examples of norms are now given.

Example 2.1.1 $d(A) = \text{Max}\{a_1, a_2, \ldots\}$

1) Clearly, $d(A) = 0$ if $A = 0$, $d(A) >$ if $A \neq 0$
2) Clearly, $d(xA) = \text{Max}\{xa_1, xa_2, \ldots\} = x\text{Max}\{a_1, a_2, \ldots\} = xd(A)$.

3) $d(A+B) = \text{Max}\{a_1 + b_1, a_2 + b_2, \ldots\} \leq \text{Max}\{a_1 + \text{Max}\{b_1, b_2, \ldots,\}a_2 + \text{Max}\{b_1, b_2, \ldots\}, \ldots\} \leq \text{Max}\{b_1, b_2, \ldots\} + \text{Max}\{a_1, a_2, \ldots\}$

Example 2.1.2 $d(A) = \sum_{n=1}^{\infty} f(n)a_n$ where f is a function from N into R.

1) since $f(n)>0$ for all n, $d(A) = 0$ if $A = 0$ and $d(A)> 0$ if $A \neq 0$.

2) $d(xA) = \sum_{n=1}^{\infty} f(n)xa_n = x \sum_{n=1}^{\infty} f(n)a_n = xd(A)$.

3) $d(A + B) = \sum_{n=1}^{\infty} f(n)\ (a_n + b_n) = \sum_{n=1}^{\infty} f(n)a_n + \sum_{n=1}^{\infty} f(n)bn = d(A) + d(B)$.

Example 2.1.2 is very important since it is the only example of norm (as shown in the next theorem) for which Condition 3 of Definition 2.1.7 is an equality. Such a norm shall be called a strong norm.

Theorem 2.1.1 The only norms for which Condition 3 is an equality are those of the form $d(A) = \sum_{n=1}^{\infty} f(n)a_n$ where f is a function from N into R.

Proof

Example 2.1.2 shows that d is a norm. Suppose d' is a norm for which Condition 3 is also an equality, If $d'(I_1), d'(I_2), \ldots$ are known, $d'(A)$ can be found for any nonzero A. This is done in the following manner: Let A be any nonzero member of L. Since there are only finitely many nonzero terms in A, there exists a last nonzero term: a_n. Now $A = a_1I_1 + a_2I_2 + \ldots + a_nI_n$ and $d'(A) = d'(a_1I_1 + a_2I_2 + \ldots + a_nI_n)$. Because Condition 3 is an equality, it follows that $d'(A) = d'(a_1I_1) + \ldots + d'(a_nI_n)$. By Condition 2, $d'(A) = a_1d'(I_1) + \ldots + a_nd'(I_n)$. This completes the proof because $d'(I_n)$ must be greater than zero in order to satisfy Condition 1.

2.2 Association Non-Negative Real Numbers With Digraphs

In this section, a method of associating non-negative extended real numbers with each digraph in a class of digraphs, Q, is developed. This number is assigned in such a way as to represent a mix of certain properties, each of which is specified by a sequence in L. To accomplish this, consider an n-tuple of functions $(h^1, \ldots h^n)$ from NXQ into I such that for all $G \in Q$ and for all $m \in N$, $h^i(m,G) = 0$ for all but finitely many m, for $i = 1, \ldots, n$. These n functions associate each $G \in Q$ with n sequences in L. Set $S^i = (h^i(1,G), h^i(2,G), \ldots)$.

Definition 2.2.1 $S = (S^1, S^2, \ldots S^n)$ is called the sequence tuple of G.

Suppose $d = (d^1, d^2, \ldots, d^n)$ is an n-tuple of norms on L and there is a subset D of L^n closed under sequence addition such that there is no digraph in Q with a sequence tuple not in D. Each digraph G Q is associated with a non-negative extended real number by means of a function f(S,D). This number may be used to order the digraphs in Q. The ordering is accomplished in the usual manner.

Definition 2.2.2 Let $G,H \in Q$ with sequences tuples $S_g = (S_g^1, \ldots, S_g^m)$ and $S_h = (S_h^1, \ldots, S_h^h)$ respectively and let $d = (d_1, \ldots, d_n)$ be an n-tuple of norms on L; then $G \leq H$ iff $f(S_g, d) \leq f(S_h, d)$.

Theorem 2.2.1 If $G,H \in Q$ have the same sequence tuples, then for all $F \in Q$, $G \leq F$ iff $H \leq F$, and $F \leq G$ iff $F \leq H$.

Proof

This follows at once from the definitions.

The notions defined so far may be used to obtain several partitions of Q. This is accomplished by defining several equivalence relations.

Definition 2.2.3 *Let G, $H \in Q$; then GS^iH iff G and H have the same i^{th} sequence in their sequence tuple.*

Definition 2.2.4 Let G, $H \in Q$; then GSH iff G and H have the same sequence tuple.

Definition 2.2.5 Let G, $H \in Q$; then GFH iff $f(S_1, d) = f(S_2, d)$, where S_1 and S_2 are sequence tuples for G and H respectively, and d is an n-tuple of norms on L of the same dimension as S_1 and S_2.

Remark The common refinement of the partitions generated by S^i equivalence is the partition generated by S equivalence.

F-equivalence will be used to resolve a slight difficulty associated with the ordering obtained in the previous definition. As it stands now, $\leq$ is not a partial ordering of Q unless f is 1-1 and for each $G \in Q$ the sequence tuple is unique. Nevertheless, it is an easy matter to show that $\leq$ is a partial ordering of the equivalence classes of f. This is evident since $\leq$ was not a partial ordering of Q in consequence of the property of antisymmetry.

Definition 2.2.6 Let P be the collection of equivalence classes of f.

Theorem 2.2.2 $\leq$ is a partial ordering of P.

Proof

1) If $S \in P$, then $S \leq S$, since any two members of S have the same f value.

2) Let $S,T \in P$. If $S \leq T$ and $T \leq S$, then S and T must have the same f value, hence S=T.

3) Let $X,Y,Z, \in P$. $X \leq Y$ imples the f value shared by members of X is less than or equal to the f value shared by the members of Y. $Y \leq X$ implies that the f value shared by members of Y is less than or equal to the f value shared by members of Z. The above statements imply the f value shared by members of X is less than or equal to the f value shared by the members of Z, hence $X \leq Z$.

Theorem 2.2.3 $(P, \leq)$ is a lattice.

Proof

Let $X,Y \in P$. At least one of the following must hold:
(1) $X \leq Y$ or (2) $Y \leq X$. Assume that 1) holds; then l.u.b. (X,y)=Y and g.l.b. (X,Y) = X. Assume that (2) holds; then l.u.b. (X,Y) = X and g.l.b. (X,Y) = Y. If both hold, l.u.b. (X,Y) = g.l.b. (X,Y) = X=Y.

Theorem 2.2.4 $(P, \leq)$ has a zero if there is a digraph $H \in Q$ such that $f(S_h,d) = 0$.

Proof

Clearly, if $Z = \{G \in Q |$ the f value of G is zero$\}$, then for all $X \in P$ l.u.b. (X,Z)=X.

Theorem 2.2.5 $(P, \underline{F})$ has an identity if there is a digraph in Q such that $f(S_h,d) = \infty$.

Proof

Clearly, if $I = \{G$ in $Q |$ the f value of G is$\infty\}$ for all X in P g.l.b.(I,X) = X.

Theorem 2.2.6 $(P, \leq)$ is a distributive lattice.

Proof

This is evident from the fact that P may be thought of as some subset of the extended real numbers with the natural order.

Theorem 2.2.7 Suppose there are digraphs $H,G \in Q$ such that $f(S_h,d) = \infty$ and $f(S_g,d) = 0$; then $(P, \leq)$ is complemented iff P consists of two classes Z and I defined as before.

Proof

If P consists of only the two classes Z and I, then $(P, \leq)$ is complemented. Suppose there are more than two classes in P. This means that there is a class A that the f value for A is not 0 or ∞. For this A there must be an A' such that g.l.b.

(A,A') = Z. The only way this may happen is for A' to be Z. But l.u.b.(A,Z) = A, not I. So (P, $\leq$) is not complemented.

The following theorem relates (P, $\leq$) to the algebra for the multiple-valued signal processing with limiting which was developed by Epstein in [1].

Theorem 2.2.8 Suppose there are digraphs H,G$\in$Q such that $f(S_h,d) = \infty$ and and $f(S_g,d)=0$; then (P, $\leq$) is identical with the algebra for multiple-valued signal processing with limiting.

Proof

By Theorem 2.2.4, Theorem 2.2.5, and Theorem 2.2.6 it is evident that (P, $\leq$) is a distributive lattice with Z and I. It must be shown that (P,$\leq$) has the following properties:

(1) For every X and Y in (P,$\leq$) there is a grestest complemental element X => Y in L satisfying g.l.b. (X, (X=> Y)) $\leq$ Y; that is, if b is a complemented element of L satisfying g.l.b. (X, b) $\leq$ Y, then b$\leq$ X => Y, with X => Y being a complemented element of L.

(2) For every X and Y in L, l.u.b. ((X => Y), (Y => X)) = 1. By the argument in Theorem 2.2.7 the only complemented elements in (P,$\leq$) are Z and I. If X $\leq$ Y, then X => Y = I. This is evident since g.l.b. (X,1) $\leq$ Y. If X > Y, then X => Y = Z. This is evident since g.l.b. (X,Z)=Z $\leq$ Y while g.l.b. (X,I) $\not\leq$ Y. So (1) holds. Since the equivalence classes of digraphs are linearly ordered either X => Y = I or Y => X = I and (2) holds.

3. APPLICATIONS OF THE GENERAL THEORY

3.1 Balance in Signed Digraphs

In this section, the methods of the Part Two will be used to develop a measure of balance in signed digraphs. The method developed by Norman and Roberts [3] will be shown to be a special case of this measure.

Intuitively, any measure of balance is signed digraphs should involve some kind of ratio between balanced semicycles and semicycles with negative signs. As a consequence, the sequence tuples will be ordered pairs and the function f will be a special kind of function.

Definition 3.1.1 f(S,d) is called a ratio function if it is equal to $d^1(S^1)/d^2(S^2)$ or $d^2(S^2)/d^1(S^1)$ where $S=(S^1,S^2)$ is a sequence tuple and $d=(d^1,d^2)$ is an order pair of norms.

For the rest of this section, only ratio functions will be considered. If a digraph has no semicycle, the question of balance is not relevant. Consequently, the case of signed digraphs that can be ordered according to balance consists of those signed digraphs will at least one semicycle.

Definition 3.1.2 Let Q be the class of all signed digraphs with at least one semicycle.

Definition 3.1.3 Let h^1, a function from NxQ into I, be defined as follows: $h^1(m,G)$ is the number of semicycles of length m + 2 whose sign is positive.

Definition 3.1.4 Let h^2, a function from NxQ into I, be defined as follows: $h^2(m,G)$ is the number of semicycles of length M + 2 whose sign is negative.

Definition 3.1.5 Let D = LxL-(0,0).

Lemma For each sequence tuple in D there is a signed digraph in Q with that sequence tuple.

Proof

Suppose (S^1,S^2) is anarbitrary sequence tuple in D. At least one of the S^i is not zero, since $(0,0)\epsilon D$. If $S^1 \neq 0$, proceed as follows: Start with a point x. For each $S^1_i \in S^1$, construct s^1_i distinct positive cycles of length i + 2 whose first and last point is x. If $S^2 \neq 0$ proceed as follows: Start with point x. For each $s^2_i \in S^2$ construct S^2_i distinct positive cycles of length i+2 whose first and last point is x. It should be noted that the definition of L insures that a structure so constructed will be finite.

Example 3.1.1 Suppose $(S^1,S^2) = ((0,1,0,\ldots),(2,1,0,\ldots))$ then the resultant graph would be:

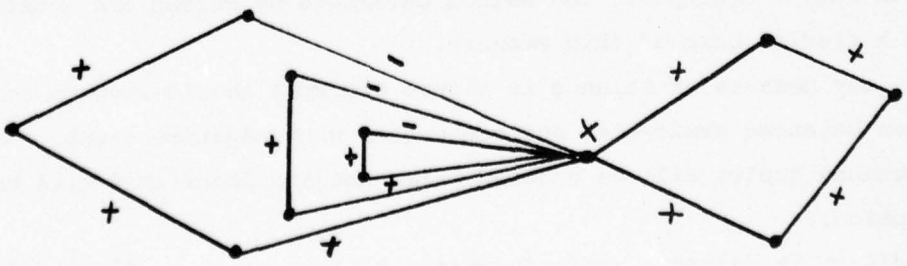


Fig. 3.1.1

Proof

Now $d^0{}_1(A) = \sum_{n=1}^{\infty} f^0{}_1(n)a_n$

$d^1{}_1(A) = \sum_{n=1}^{\infty} f^1{}_1(n)a_n$

$d^0{}_2(A) = \sum_{n=1}^{\infty} f^0{}_2(n)a_n$

$d^1{}_2(A) = \sum_{n=1}^{\infty} f^1{}_2(n)a_n$

where $f^0{}_1$, $f^1{}_1$, $f^0{}_2$, $f^1{}_2$ are functions from N into R since $d^0{}_1$, $d^1{}_1$, $d^0{}_2$, $d^1{}_2$ are all strong norms. In order to prove the theorem, it will be shown that $f^0{}_1(n) = pf^0{}_2(n)$ and $f^1{}_1(n) = qf^1{}_2(n)$ for all n. Suppose n is given. For any k and t, it is a simple matter to construct a signed digraph with sequence tuple (kI_n, tI_1). It is also easy to construct a digraph H with sequence tuple (I_1, I_1). Now $G \leq H$ iff

$$\frac{d^0{}_1(kI_n)}{d^1{}_1(tI_1)} \leq \frac{d_1(I_1)}{d^1{}_1(I_1)} \quad \text{iff} \quad \frac{kf^0{}_1(n)}{tf^1{}_1(1)} \leq \frac{f^0{}_1(1)}{f^1{}_1(1)} \quad \text{iff} \quad \frac{f^0{}_1(n)}{f^0{}_1(n)} \leq \frac{t}{k}$$

But since $(d^0{}_1, d^1{}_1)$ and $(d^0{}_2, d^1{}_2)$ induce the same order on Q it is evident that $G \leq H$ iff $f^0{}_2(n)/f^0{}_2(1) \leq t/k$. Since this happens for all t and k, it is clear that:

$$\frac{f^0{}_1(n)}{f^0{}_1(1)} = \frac{f^0{}_2(n)}{f^0{}_2(1)};$$

hence $f^0{}_1(n) = [f^0{}_1(1)/f^0{}_2(1)]f^0{}_2(n)$. A similar argument shows:

$$f^1{}_1(n) = [f^1{}_1(1)/f^1{}_2(1)]\, f^1{}_2(n)$$

In view of this last theorem, it is evident that the order derived from the digraphs in Q is quite dependent on the choice of d. From Theorem 2.1.1, it is evident that $d^0(A) = \sum_{n=1}^{\infty} g^0(n)a_n$ and $d^1(A) = \sum_{n=1}^{\infty} g^1(n)a_n$, where g^0 and g^1 are functions from N into R. This being the case, d^0 and d^1 may be described in terms of g^0 and g^1. It would be reasonable to choose both g^0 and g^1 as some kind of decreasing functions,

since intuitively it would seem that the shorter the semicycle, the more weight it should be given in determining balance.

The order developed in Norman and Roberts [3] is a special case of the techniques developed here. If d^0 and d^1 are identical, this order is obtained.

3.2 Cost of Retrieval From a Binary Tree

Palmer, Rahimi, and Robinson [4] in a paper dealing with the efficiency of binary tree storage, develop a measure for the average and variance of the number of comparisons needed to retrieve one item from a store of n items. In this section, their results will be shown to be related to the class of measures derived in Part Two of this paper.

Definition 3.2.1 Let B be the class of all binary forests.

For each $G \in B$, associate the following sequence: $A = (a_1, a_2, \ldots)$ where a_i is the number of paths on length i. As binary forests are finite, it follows that a_i will be zero for all but a finite number fa_i.

Definition 3.2.2 The average length μ of a binary forest with sequence $A = (a_1, a_2, \ldots)$ is

$$\sum_{i=1}^{\infty} ia_i \Big/ \sum_{i=1}^{\infty} a_i$$

Definition 3.2.3 The variance σ^2 of a binary forest with sequence $A = (a_1, a_2, \ldots)$ is

$$\sum_{i=1}^{\infty} (ia_i - \mu a_i)^2 \Big/ \sum_{i=1}^{\infty} a_i.$$

Both these measures reduce to the results given by Palmer, Rahimi, and Robinson [4] when the particular forest in question is the forest made up on all possible binary stores of n items. Palmer, Rahimi, and Robinson [4] were able to express their results in a manner which showed that the variance approaches $7-2/3\pi^2$ for large n. This raises the rather interesting question as to when techniques similar to theirs may be used in other instances of graph-theoretic measurement.

Another way in which the theory developed in part two of this paper may be applied to the results derived by Palmer, Rahini, and Robinson [4] is to let B be the class of all binary forests in which each tree has exactly n nodes. The theory then provides a means of ordering the various forests according to average length and

variance. This ordering might be of value in devising a scheme for merging representatives from the various classes.

3.3. A Measure of Transitivity

The question of deciding how transitive a particular finite asymmetric digraph is of great value in trying to measure the reliability of judges in comparison tests. Until now, no satisfactory quantitative measure has been developed for this. The theory developed in the second part of this paper lends itself readily to this problem. The only restriction made upon the finite asymmetric digraphs is that they contain at least one path or length two.

Definition 3.3.1 Let Q be the class of all finite asymmetric digraphs with at least one path of length two.

Definition 3.3.2 If $G \in Q$ and $S \leq V(G)$, then $\langle s \rangle$ is a maximal induced weakly connected intransitive subgraph iff it is weakly connected and intransitive and there is no point $v \in V(S)-S$ such that $\langle SU\{v\} \rangle$ is weakly connected and intransitive.

Example 3.3.1 Consider the following digraph:

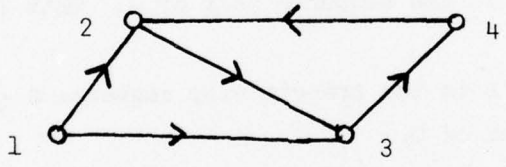


Fig. 3.3.1

let $S=\{2,3,4\}$ then S is the following digraph:

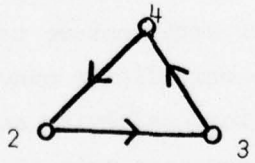


Fig. 3.3.2

Now $\langle S \rangle$ is clearly intransitive and weakly connected. It is also maximal, since if $\{1\}$ is added, the resulting digraph $\langle S \cup \{1\} \rangle$ is not intransitive. For all digraphs $G \in Q$ and for all $m \in N$, let $h(m,G)$ be the number of maximal induced weakly connected intransitive subgraphs having $m + 2$ points.

Definition 3.3.3 If $G \in Q$ and $S \leq V(G)$, the $\langle S \rangle$ is the maximal induced weakly connected

transitive subgraph iff it is weakly connected and transitive and there is no point $v \in V(G) - S$ such that $\langle S \cup \{v\}\rangle$ is weakly connected and transitive.

Example 3.3.2 Consider the digraph of the previous example. Let $S = \{1,2,3\}$ then $\langle S\rangle$ is the following digraph:

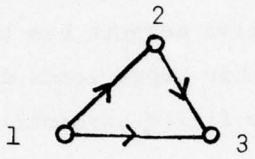


Fig. 3.3.3

Now $\langle S\rangle$ is clearly transitive and weakly connected. It is also maximal since if $\{4\}$ is added the resulting digraph, $S \cup 4$, is not transitive. For all graphs $G \in Q$ and for all $m \in N$, let $h(m,G)$ be the number of maximal induced weakly connected transitive subgraphs having $m + 2$ points.

Now let $D = L \times L - (0,0)$. There is no digraph in Q such that its sequence pair does not belong to D. It is evidnet that for each sequence pair $(S,\bar{S})$ in D there is a digraph G in Q such that $(S,\bar{S})$ is the sequence pair of G. This is proved in the following theorems.

Theorem 3.3.1 Every member of L is the transitivity sequence S [intransivity sequence $\bar{S}$] of some member of Q.

Proof

Let $S = (a_1,a_2,\ldots)$ be the sequence in L. For each nonzero term a_i make a_i copies of the transitive tournament on $i + 2$ points. Since there are only finitely many nonzero tersm, the resulting structure is a digraph. S will be its transivity sequence. Let $\bar{S} = (b_1,b_2,\ldots)$ be a sequence in L. For each nonzero term b_i make b_i copies of the cycle in $i + 2$ points. Since there are only finite many nonzero terms, the resulting structure is a digraph. $\bar{S}$ will be its intransitivity sequence.

Corollary Each pair $(S,\bar{S}) \in D$ is the sequence pair of some digraph in Q.

Proof

Combine the constructions in the proof of Theorem 3.3.1 and note that $(0,0) \in D$.

Let f be any ratio function and d and d' two strong norms on L. Q may be ordered by its f-values. By the results in the previous chapter, the following theorems are evident. Let G_1, G_2, $\in Q$; then $G_1 F G_2$ iff $f(S_1,\bar{S}_1,d,d') = f(S_2,\bar{S}_2,d,d')$,

where $(S_1,\bar{S}_1)$ and $(S_2,\bar{S}_2)$ are sequence pairs of G_1 and G_2 respectively.

Let W be the collecgion of equivalence classes of F.

Theorem 3.3.2 $(W,\leq)$ is a distributive lattice.

Theorem 3.3.3 $(W,\leq)$ is not complemented.

In consequence of the results of Part Two of this paper it is evident that the order derived for digraphs in Q is dependent on the choice of d and d'. It is evident that

$$d(S)= \sum_{n=1}^{\infty} g(n)s_n \text{ and } d'(S) = \sum_{n=1}^{\infty} g'(n)s_n ,$$

where g and g' are functions from N into R. This being the case, d and d' may be described in terms of g and g'. It would seem reasonable to choose both g and g' as some kind of increasing function. Perhaps, $g(m) = m^p$, for all m. For example that the intransitivity of a digraph; then the following definitions could be made:

$$d(A)= \sum_{n=1}^{\infty} n^2 a_n \text{ and } d'(A)= \sum_{n=1}^{\infty} n\, a_n \quad .$$

The function in this case would be $f(S,\bar{S},d,d') = d'(S)/d'(\bar{S})$.

3.4 A Measure of Symmetry

Another property of digraphs which might be of some interest to a psychologist of sociologist is that of symmetry.

Example 3.4.1 Suppose there are 5 couples in an apartment complex. A psychologist might ask each couple to list their friends. The psychologist might then represent their responses by the following digraph:

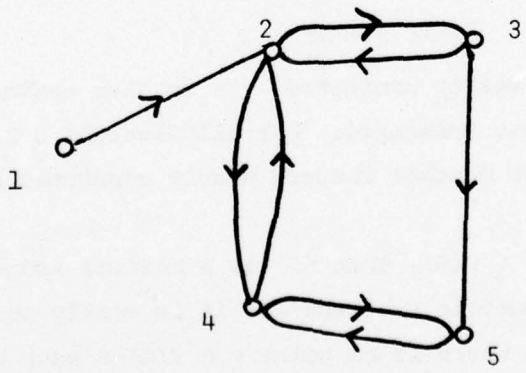


Fig. 3.4.1

A natural question to ask would be how symmetric is the digraph? Symmetry in this case would indicate a friendship was mutual.

Definition 3.4.1 Let P be the class of all finite digraphs with a least one edge.

Definition 3.4.2 If $G \in P$ and $S \leq V(G)$, then $\langle S \rangle$ is a maximal induced weakly connected symmetric subgraph iff it is weakly connected and symmetric and there is no point $v \in V(G) - S$ such that $\langle S \cup \{v\} \rangle$ is weakly connected and symmetric.

Example 3.4.1 Consider the following digraph:

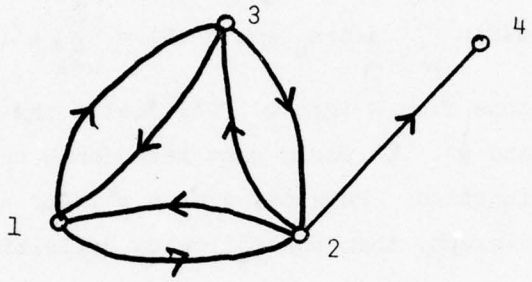


Fig. 3.4.2

Let $S = \{1,2,3\}$; then $\langle S \rangle$ is the following digraph:

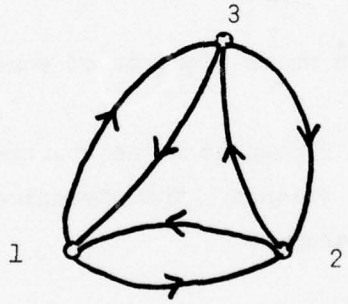


Fig. 3.4.3

Now $\langle S \rangle$ is clearly symmetric and weakly connected. It is also maximal since if 4 is added, the resulting digraph is not symmetric. For all digraphs $G \in P$ and for all $m \in N$, let $h(m,G)$ be the number of maximal induced weakly connected symmetric subgraphs having $m + 1$ points.

Definition 3.4.3 If $G \in P$ and $S \leq V(G)$, then $\langle S \rangle$ is a maximal induced weakly connected asymmetric subgraph iff it is weakly connected and asymmetric and there is no point $v \in V(G)-S$ such that $\langle S \cup \{v\} \rangle$ is weakly connected and asymmetric.

Definition 3.4.4 For all digraphs $G \in P$ and for all $m \in N$, let $\bar{h}(m,G)$ be the number of maximal induced weakly connected asymmetric subgraphs having $m + 1$ points.

Now let $D = LxL - (0,0)$. There is no digraph in P such that its sequence pair

does not belong to D. For each sequence pair $(S,\bar{S}) \in D$ there is a digraph $G \in P$ such that $(S,\bar{S})$ is the sequence pair of G. This is evident from the following theorem:

Theorem 3.4.1 Every member of L is the symmetry sequence S [asymmetry sequence $\bar{S}$] of some member of Q.

Proof

Let $S = (s_1,s_2,\ldots)$ be a sequence in L. For each nonzero term s_i makes s_i copies of the digraph for which $D(G) = V(G)xV(G)$, and for which $|V(G)|=i+1$. Since there are only finitely many nonzero terms, the resulting structure is a digraph. S will be its symmetry sequence. Its asymmetry sequence will be 0. Let $\bar{S} = (s_1,s_2,\ldots)$ be a sequence in L. For each nonzero term $\bar{s}_i$ make $\bar{s}_i$ copies of a tournament haveing i + 1 points. Since there are only finitely many nonzero terms, the resulting structure is a digraph. S will be its asymmetry sequence. Its symmetry sequence will be 0.

Corollary Each pair $(S,\bar{S}) \in D$ is the sequence pair of some diagraph in P.

Proof

Combine the constructions in the proof of the above theorem and note that $(0,0) \in$ in D.

Let f be any ratio function and d and d' be strong norms on L. P may be ordered by its f-values. By the results in the last chapter, the following theorems are evident. Let $G_1,G_2 \in P$; then G_1FG_2 is and only if

$$f(S_1,\bar{S}_1,d,d') = f(S_2,S_2,d,d'),$$

where $(S,\bar{S}_1)$ and $(S_2,\bar{S}_2)$ are the sequence pairs of G_1 and G_2 respectively.

Let W be the collection of equivalence classes of F.

Theorem 3.4.2 $(W,\leq)$ is a distributive lattice.

Theorem 3.4.3 $(W,\leq)$ is not complemented.

4. DIGRAPH MEASURES, FUZZY SETS, AND N-VALUED LOGIC

Fuzzy sets were introduced by Zadeh in [5]. Intuitively, a fuzzy set is a class of objects with a continuum of grades of membership.

Definition 4.1 A fuzzy set A in X is characterized by a membership function $f_A(x)$ which associates with each point in X a real number in the interval [0,1], with the value of $f_A(x)$ at X representing the "grade of membership" of x in A.

In Part Two of this paper, a method is developed for associating an extended positive real number, say x, with each digraph G in a class of digraphs D. If $f(G) = 1 - \frac{1}{1+x}$ for each graph G in D, then this extended positive real number has been used to define a fuzzy set in G. Hence, the theory developed in Part Three may be

used to define the fuzzy set of balanced singed digraphs, the fuzzy set of transitive digraphs, and the fuzzy set of symmetric digraphs. The following theorem provides the justification for doing this.

Theorem 4.1 Let f be defined as in Part Two of this paper, D be a class of digraphs, H and K be in D, and $g_D(G) = 1 - \frac{1}{1+f(S_g,d)}$ then $H \leq K$ iff $g_D(H) \leq g_D(K)$.

Proof

If $f(S_h,d)=x$ and $f(S_k,d)=y$, then $H \leq K$ implies $x \leq y$. Since both x and y are positive, it is evident that $\frac{1}{x+1} \geq \frac{1}{y+1}$. Consequently, $1 - \frac{1}{x+1} \leq 1 - \frac{1}{y+1}$ hence $g_D(H) \leq g_D(K)$. Suppose $g_D(H) \leq g_D(K)$, this implies that $1-\frac{1}{f(S_h,d)+1} \leq 1- \frac{1}{f(S_k,d)+1}$. Consequently, $\frac{1}{f(S_h,d)+1} \leq \frac{1}{f(S_k,d)+1}$. Since both f(H) and f(K) must be nonnegative, $f(H) \leq f(K)$ and so $H \leq K$.

It is not meaningful to speak of a point x "belonging" to a fuzzy set A except in the trivial sense of $f_A(x)$ being positive. Zadeh points out in [5] that one may introduce two levels α and β $(0<\beta<\alpha<1)$ and agree to say that (1) x belongs to A if $f_A(x) \geq \alpha$, (2) x does not belong to A if $f_A(x) \leq \beta$, (3) x has an indeterminate status relative to A if $\beta < f_A(x) < \alpha$. This leads to a three-valued logic. Clearly this same strategy may be employed to obtain an n-valued logic.

REFERENCES

[1] G. Epstein, "Multiple-Valued Signal Processing with Limiting", Proceedings of of the 1972 Symposium on Multiple-Valued Logic Design, Buffalo, N.Y. (May, 1972)

[2] F. Harary, R.Z. Norman and D. Cartwright, Structural Models: An Introduction to the Theory of Directed Graphs, Wiley, New York, (1965).

[3] R.Z. Norman and F.S. Roberts, "A Derivation of a Measure of Relative Balance for Social Structures and a Characterization of Extensive Ratio Systems," J. of Math. Psych. 9 (1972).

[4] E.M. Palmer, M. Rahimi and R.W. Robinson, "Efficiency of a Binary Comparison Storage Technique." J. of A.C.M. Vol. 121, No. 3 (July 1974).

[5] L.A. Zadeh, "Fuzzy Sets", Information and Control 8, 338-353 (1965).

SUMMARY

BINARY AND MULTIPLE-VALUED MODELS OF BINARY GATE NETWORKS§

Michael Yoeli
Technion-Israel Institute of Technology
Israel

Phenomena such as hazards, races, and oscillations in binary switching circuits may be studied by means of binary, ternary, as well as various other multiple-valued mathematical models. In our paper we present and evaluate some of these models, extending earlier research in this area ([1], [2], [3]).

Let L be a finite set (the set of all possible logic values, e.g. $L = \{0,1\}$, $L = \{0, 1/2, 1\}$, etc.). We assume a gate network which has n external L-valued inputs $x_1,\ldots,x_n$ and consists of s gates $G_1,\ldots,G_s$. For gate G_j , the output is $y_j \in L$ and the gate performs a function $f_j: L^{n+s} \to L$ on the $n+s$ variables $x_1,\ldots,x_n,y_1,\ldots,y_s$.

For $j = 1,\ldots,s$, let

$$Y_j = f_j(x_1,\ldots,x_n,y_1,\ldots,y_s) \quad .$$

The ordered s-tuple $(y_1,\ldots,y_s)$ represents the present gate-state of of the network and the ordered s-tuple $(Y_1,\ldots,Y_s)$ represents the excitation for the total state $(x_1,\ldots,x_n;y_1,\ldots,y_s)$.

Now for each $x \in L^n$ define a binary relation R_x on L^s as follows.

Let $(x;y)$ be some total state of the network and Y the corresponding excitation state. If $y = Y$, then yR_xy . Otherwise, consider each i such that $y_i \neq Y_i$. Then $yR_xy^{(i)}$, where $y^{(i)}$ is obtained from y by replacing its i-th component y_i by Y_i .

The behavior of the network can now be determined by means of the relation diagram for R_x . The nodes of this diagram correspond to the elements of L^s and a directed edge is drawn from y to $\bar{y}$ iff $yR_x\bar{y}$. If yR_xy , then the total state $(x;y)$ is stable. Otherwise, follow all directed paths in the relation diagram for R_x , starting with node y . Any such path must reach a cycle after a finite number of steps. Starting from y , if we reach more than one cycle we have a critical race†. If only one cycle can be reached, and this cycle is of length greater than 1,

§This paper will appear in the book Modern Uses of Multiple-Valued Logic.

we have an oscillation. If only one cycle can be reached and this cycle is of length 1, the corresponding node gives the unique stable gate-state that the network will reach from the total state $(x;y)$, provided the input x is not changed.

If we set $L = \{0,1\}$, we obtain a binary model by means of which hazards, races and oscillations may be detected.

In [4] a theory of static hazards was developed, using a ternary algebra with $L = \{0, 1/2, 1\}$. In [5] the use of this algebra was extended to the detection of logic hazards, races, and oscillations in binary gate networks. On the one hand the approach of [5] was computationally efficient, on the other hand no precise mathematical formulation of this approach was available and consequently the results of such a ternary analysis were not always clearly understood [6]. Our paper provides a precise formulation of the ternary analysis approach, by means of the above multiple-valued network model, with $L = \{0, 1/2, 1\}$.

Namely, with every Boolean function $f:\{0,1\}^n \to \{0,1\}$ we associate its ternary extension $f^*:\{0, 1/2, 1\}^n \to \{0, 1/2, 1\}$ in the rather obvious way. Similarly, we associate with every binary gate network N a ternary gate network N^*, and then show precisely, how the properties of N relate to those of N^*. This theory indicates clearly, why the 3-valued analysis is more efficient, but also why essential information may be lost by this analysis.

The remainder of the paper surveys and evaluates other binary and multiple-valued models which have been proposed for the purpose of providing a realistic analysis of binary gate networks.

REFERENCES

[1] J.A. Brzozowski and M. Yoeli, Digital Networks, Prentice-Hall; to appear 1975.

[2] J.A. Brzozowski and M. Yoeli, "Models for Analysis of Races in Sequential Networks", to appear in Proc. 3rd MFCS Symposium, Jadwisin-Warsaw, June, 1974; Springer-Verlag.

[3] W. Assmus and M. Yoeli, "Binary and Ternary Simulations of Asynchronous Networks", Report No. 54, Institut für Theorie der Automaten und Schaltnetzwerke, GMD, Bonn; 1972.

[4] M. Yoeli and S. Rinon, "Application of Ternary Algebra to the Study of Static Hazards", J. Ass. Comp. Mach., vol. 11, pp. 84-97, Jan., 1964.

[5] E.B. Eichelberger, "Hazard Detection in Combinational and Sequential Switching Circuits", IBM J. Res. Develop., vol. 9, pp. 90-99, March, 1965.

[6] M.A. Breuer, "A Note on Three-Valued Logic Simulation", IEEE Trans. Comp., vol. C-21, pp. 399-402, April, 1972.

---

† We assume that the output coincides with the gate-state.

A TERNARY ALGEBRA FOR PROBABILITY COMPUTATION OF DIGITAL CIRCUITS

Sung C. Hu
Cleveland State University
Ohio, U.S.A.

Abstract

This paper presents a ternary algebraic approach of computing the logic circuit reliability. The main advantage of this technique is its ability to compute simply not only the reliability of the network but also the probabilities of the network producing a logic-1 and logic-0 fault. This is particularly important in applications where one type of fault is much more critical than the other.

A ternary algebra is defined and the ternary algebraic expressions for various logic gates are derived. The probability expressions are shown to directly correspond to the ternary expressions. The reliabilities of four types of uniform second order networks are analyzed and compared by using the method presented in the paper.

The technique can be easily followed by logic designers not familiar with more sophisticated reliability theory. This work can be extended to the study of circuits with n different failure modes by using (n + 1)-valued algebra.

I. INTRODUCTION

A failure in a logic gate may be attributed to any failure in the input or the output circuitry. A logic gate whose input and output components are subject to certain failure probabilities will be referred to as a probablistic logic gate. A signal in a probablistic digital circuit may attain an erroneous logic-1 value or an erroneous logic-0 value due to different types of component failures. Whether the failure is permanent or temporary, an erroneous logic-1 (0) signal will be referred to as a logic-1 (0) fault. That is, a logic-1 (0) fault is defined as one where the signal assumes the logic-1 (0) value when it should be at the logic-0 (1) value.

Chen [1] considered the use of Boolean algebra for reliability estimation of probablistic switching circuits in which a component may either be functioning or failing. Eichelberger [2] and Yoeli and Rinon [3] suggested the use of ternary algebra to the study of hazards by using the third value ($\frac{1}{2}$) for hazardous conditions. Multivalued algebra has also been used in other studies of binary switching circuits [4,5]. It is proposed in this paper that ternary algebra be applied to the study of reliability analysis problem when two different types of circuit failures are considered. This work can be extended to the study of circuits with n different failure modes by using (n + 1)-valued algebra.

II. TERNARY ALGEBRA

The ternary algebra to be used in this paper is based on the set $T = \{0, \frac{1}{2}, 1\}$. The values 0 and 1 will be interpreted as logic-0 and logic-1 faults respectively. (Alternatively, they may be interpreted as open and short circuit failures.) The value $\frac{1}{2}$ will be interpreted to represent fault-free operations. Three ternary algebraic operations, AND ($\cdot$), OR (+), and CYCLING ('), are defined in Fig. 1. Note that the AND and OR

·	0	$\frac{1}{2}$	1
0	0	0	0
$\frac{1}{2}$	0	$\frac{1}{2}$	$\frac{1}{2}$
1	0	$\frac{1}{2}$	1

(a)

+	0	$\frac{1}{2}$	1
0	0	$\frac{1}{2}$	1
$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	1
1	1	1	1

(b)

A	A'
0	$\frac{1}{2}$
$\frac{1}{2}$	1
1	0

(c)

Fig. 1. Ternary algebraic operations: (a) AND, (b) OR, and (c) CYCLING

operations are simple extensions of the Boolean AND and OR. The AND symbol (·) is often omitted in an algebraic expression. It can be shown that the AND operation together with the CYCLING operation form a functionally complete set [6] . However, other operations are often used in order to keep the ternary algebraic expressions simple. The operations NEGATION ($\bar{}$) and EXCLUSIVE-OR ($\oplus$) as defined below will be used repeatedly in the subsequent discussions.

$$\left.\begin{aligned} &\text{NEGATION OF } A = \bar{A} = A' + A'' \\ &\text{EXCLUSIVE-OR } (A,B) = A \oplus B = A\bar{B} + \bar{A}B \end{aligned}\right\} \text{Where } A, B \in T.$$

Note that the definition for the NEGATION operation is not a conventional one. Under this definition, the law of involution is not true although the laws of deMorgan still hold. That is

$$\bar{\bar{A}} \neq A,$$
$$\overline{A + B} = \bar{A} \cdot \bar{B}, \quad \overline{A \cdot B} = \bar{A} + \bar{B}$$

Both NEGATION and EX-OR operations as defined here are for the convenience of computing probabilities. This will become clear after reading Section IV. Other algebraic properties that will be useful later are given below.

$$A + B = A \oplus \bar{A}B = B \oplus \bar{B}A$$
$$A(B \oplus C) = AB \oplus AC$$
$$AB \oplus A\bar{B} = A$$
$$AB \oplus AB' \oplus A\bar{B}'' = A$$
$$AB \oplus AB' = A\bar{B}''$$

The hierarchy of operation is, in decreasing order, CYCLING, NEGATION, AND, and OR (EX-OR).

III. RELIABILITY MODELING OF LOGIC GATES

Assuming all component failures are independent and all failure modes are mutually exclusive, a probablistic logic gate may be modeled as a perfect (fault-free) gate connected to probablistic input and output blocks as shown in Fig. 2. These blocks are to be considered as having switches that are able to switch the line to any one of the ternary values. That is, the probablistic nature of the input and output components are taken out of the gate and represented in separate blocks. Each block, therefore, will have associated with it three probability figures: the probability of committing a logic-1 fault, the probability of committing a logic-0 fault, and the probability of operating correctly. Throughout this paper, small case x's and z will be used to represent input and output blocks while capital X's and Z are for input and output signals. If the

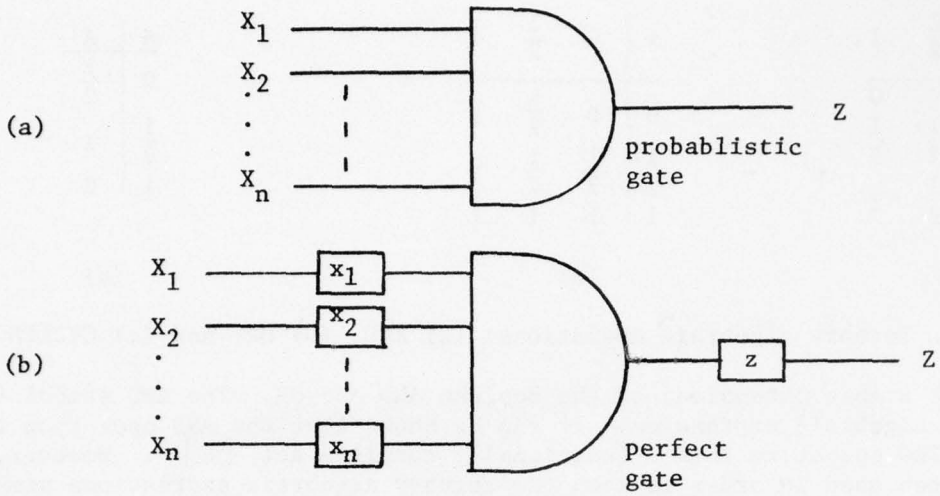


Figure 2. (a) Probablistic n-input AND gate.
(b) Probablistic model of an n-input AND gate.

input signals are assumed to be error-free, then whether Z is error-free or not is determined by z, and x_1, x_2, . . ., x_n. This relationship can be easily represented in a ternary truth table. Figure 3 shows such a table for a two-input AND gate. Note that

x_1	x_2	z	Z
0	0	0	0
0	0	½	0
0	0	1	1
0	½	0	0
0	½	½	0
0	½	1	1
0	1	0	0
0	1	½	0
0	1	1	1

x_1	x_2	z	Z
½	0	0	0
½	0	½	0
½	0	1	1
½	½	0	0
½	½	½	½
½	½	1	1
½	1	0	0
½	1	½	d
½	1	1	1

x_1	x_2	z	Z
1	0	0	0
1	0	½	0
1	0	1	1
1	½	0	0
1	½	½	d
1	½	1	1
1	1	0	0
1	1	½	1
1	1	1	1

Figure 3. Ternary truth table of a probablistic two-input AND gate.

when z is either 0 or 1, Z follows z. When $z = ½$, $Z = x_1 \cdot x_2$. The two d's in the Z column represent indeterminant states. They may be interpreted either as a fault or a correct state depending on the assumptions made. If a more conservative estimate of the reliability is desired, the d's are to be treated as one of the fault states (1 in the case of an AND gate). This is equivalent to the assumption that the output is faulty when one or more inputs are faulty regardless of the logic operation involved. It represents the lower bound on the reliability estimation of the circuit. This assumption is used in all subsequent derivations. One way to compute the probability of $Z = ½$ (or 0, or 1) is to compute the probabilities of each combination in the ternary truth table and then add the probabilities corresponding to rows that gives $Z = ½$ (or 0, or 1). However, much computational time may be saved if ternary expressions are obtained. This will be discussed in the next section.

AD-A045 757 INDIANA UNIV BLOOMINGTON DEPT OF COMPUTER SCIENCE F/G 9/2
PROCEEDINGS OF THE 1975 INTERNATIONAL SYMPOSIUM ON MULTIPLE-VAL--ETC(U)
MAY 75 G EPSTEIN N00014-75-C-0449
UNCLASSIFIED MVL-75-001 NL

4 OF 6
AD A045757



The probability model shown in Figure 2 assumes fault-free input signals. If the probabilities of input signals are considered, the model shown in Figure 4 would be used. Blocks $b_1, \ldots, b_n$ represent the probabilities associated with the input signals $X_1, \ldots, X_n$. Note that when two probablistic blocks are connected in series, the output has a logic-0 (1) fault if the block closer to the output has a logic-0 (1) fault or if the block closer to the output works properly and the other block has a logic-0 (1) fault. The output is correct only if both blocks operate properly. Thus, this model can be easily converted to the earlier model by simply combining blocks b_i with x_i. This, in effect, modifies the probability values of the blocks $x_i, \ldots, x_n$. If these modified blocks are named $y_1, \ldots, y_n$, then

$$P[y_i = 0] = P[x_i = 0] + P[x_i = \tfrac{1}{2}]\, P[b_i = 0]$$

$$P[y_i = \tfrac{1}{2}] = P[x_i = \tfrac{1}{2}]\, P[b_i = \tfrac{1}{2}]$$

$$P[y_i = 1] = P[x_i = 1] + P[x_i = \tfrac{1}{2}]\, P[b_i = 1]$$

Therefore, whether the input signals are probablistic or not, the only difference is the probability values used in the x blocks of Figure 2.

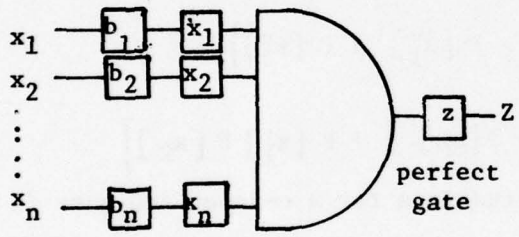


Figure 4. Probablistic model of a probablistic n-input AND gate with probablistic input signals.

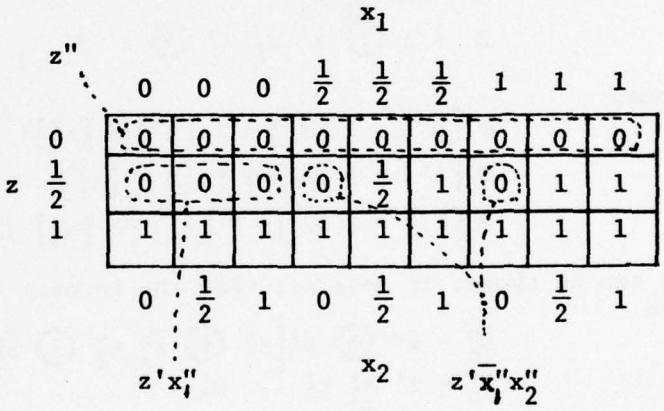


Figure 5 Ternary map of a probablistic two-input AND gate.

IV. TERNARY EXPRESSIONS FOR PROBABLISTIC LOGIC GATES

It is obvious from the definitions of ternary operations that the following correspondences between ternary operations and probability computations can be established.

$$(1)\quad \begin{cases} Z = x_1 \cdot x_2 \\ P[Z] = P[x_1] \cdot P[x_2] \end{cases}$$

$$(2)\quad \begin{cases} Z = \bar{x} \\ P[Z] = P[\bar{x}] = 1 - P[x] \end{cases}$$

Note that A $\oplus$ B implies the union of A and B is empty. It is known that the probability of the union of disjoint events is the sum of the individual probabilities [7]. Hence,

$$(3)\ \begin{cases} Z = x_1 \oplus x_2 \\ P[z] = P[x_1] + P[x_2] \end{cases}$$

$$(4)\ \begin{cases} Z = x_1 + x_2 = x_1 \oplus \bar{x}_1 x_2 \\ P[Z] = P[x_1] + (1-P[x_1])\, P[x_2] = P[x_1] + P[x_2] - P[x_1]\, P[x_2] \end{cases}$$

With these correspondences, the probability computation is very straightforward once the ternary expressions are obtained. It is evident, though, that the $\oplus$-operation is a much preferred operation to the + -operation when probability computation is the main objective. Ternary expressions for probablistic logic gates may be obtained from their ternary truth tables if an uncycled variable is used to represent 1, a single-cycled variable for $\frac{1}{2}$, and a double-cycled variable for 0. The ternary expressions for the two-input AND-gate are shown below and are obtained with the help of a ternary map. Note that when $\oplus$ -operation is used, each cell is allowed to be included in only one term. Figure 5 shows the grouping for Z".

$$Z'' = z'' \oplus z' x_1'' \oplus z' \bar{x}_1'' x_2'' = z'' \oplus z' (x_1'' \oplus \bar{x}_1'' x_2'')$$
$$Z' = z' x_1' x_2'$$
$$Z = z \oplus z' x_1 \bar{x}_2'' \oplus z' x_1' x_2 = z \oplus z' (x_1 \bar{x}_2'' \oplus x_1' x_2)$$

Thus,

$$P[Z''] = P[z''] + P[z']\left\{P[x_1''] + (1 - P[x_1'']\ )\, P[x_2'']\right\}$$
$$P[Z'] = P[z']\, P[x_1']\, P[x_2']$$
$$P[Z] = P[z] + P[z']\left\{P[x_1]\,(1 - P[x_2'']\ ) + P[x_1']\, P[x_2]\right\}$$

It can be shown, in general, that the ternary expressions for a n-input AND-gate ($n \geq 2$) are

$$Z_n'' = z'' \oplus z'\left[x_1'' \oplus \bar{x}_1'' x_2'' \oplus \bar{x}_1'' \bar{x}_2'' x_3'' \oplus \cdots \oplus (\bar{x}_1'' \cdots \bar{x}_{n-1}'') x_n''\right]$$
$$Z_n' = z' x_1' x_2' \cdots x_n'$$
$$Z_n = z \oplus z' A_n$$

where $A_n = A_{n-1} \bar{x}_n'' \oplus x_1' x_2' \cdots x_{n-1}' x_n$,

and $A_1 = x_1$.

Since the OR-gate is the "dual gate" of the AND-gate, all one has to do to obtain the ternary expressions for OR gates is to interchange uncycled variables with double-cycled variables and vice versa for all variables including Z. For example, a three-input OR gate would have the following ternary expressions.

$$Z_3'' = z'' \oplus z'\left[(x_1'' \bar{x}_2 \oplus x_1' x_2'')\, \bar{x}_3 \oplus x_1' x_2' x_3''\right]$$
$$Z_3' = z' x_1' x_2' x_3'$$
$$Z_3 = z \oplus z' (x_1 \oplus \bar{x}_1 x_2 \oplus \bar{x}_1 \bar{x}_2 x_3)$$

NAND-gates are inverted-input OR gates. Therefore, the ternary expressions for the NAND gates may be obtained by changing input variables x to x" and x" to x. The following are the expressions for a three-input NAND gate.

$$Z_3'' = z'' \oplus z' \left[(x_1 \bar{x}_2'' \oplus x_1' x_2) \bar{x}_3'' \oplus x_1' x_2' x_3 \right]$$
$$Z_3' = z' x_1' x_2' x_3'$$
$$Z_3 = z \oplus z' (x_1'' \oplus \bar{x}_1'' x_2'' \oplus \bar{x}_1'' \bar{x}_2'' x_3'')$$

Similarly, ternary expressions for NOR gates may be obtained by interchanging input variables x and x" of those for AND gates. Table I gives the summary of ternary expressions for various n-input logic gates.

Table I Ternary Expressions for n-Input Probablistic Logic Gates

Gate Type	Z"	Z'	Z
AND	$z'' \oplus z' C_n$	$z' \prod_{i=1}^{n} x_i'$	$z \oplus z' A_n$
OR	$z'' \oplus z' B_n$	$z' \prod_{i=1}^{n} x_i'$	$z \oplus z' D_n$
NAND	$z'' \oplus z' A_n$	$z' \prod_{i=1}^{n} x_i'$	$z \oplus z' C_n$
NOR	$z'' \oplus z' D_n$	$z' \prod_{i=1}^{n} x_i'$	$z \oplus z' B_n$

Where n = number of inputs

$$A_n = A_{n-1} \bar{x}_n'' \oplus \left[\prod_{i=1}^{n-1} x_i' \right] x_n, \qquad A_1 = x_1$$

$$B_n = B_{n-1} \bar{x}_n \oplus \left[\prod_{i=1}^{n-1} x_i' \right] x_n'', \qquad B_1 = x_1''$$

$$C_n = x_1'' \oplus \bar{x}_1'' x_2'' \oplus \bar{x}_1'' \bar{x}_2'' x_3'' \oplus \cdots \oplus (\bar{x}_1'' \bar{x}_2'' \cdots \bar{x}_{n-1}'') x_n''$$

$$D_n = x_1 \oplus \bar{x}_1 x_2 \oplus \bar{x}_1 \bar{x}_2 x_3 \oplus \cdots \oplus (\bar{x}_1 \bar{x}_2 \cdots \bar{x}_{n-1}) x_n$$

V. SIMPLIFICATION OF PROBABLISTIC EXPRESSIONS

It is reasonable to assume that all input blocks of a gate will have identical probabilities. If so, let

p = P [an input circuit produces a logic-1 fault]
q = P [an input circuit produces a logic-0 fault]
r = P [an input circuit operates correctly]
s = P [the output circuit produces a logic-1 fault]
t = P [the output circuit produces a logic-0 fault]
u = P [the output circuit operates correctly]

Then, for a n-input AND gate, the probability expressions are

$$P[Z''] = t + u[1 - (1 - q)^n]$$
$$P[Z'] = ur^n$$
$$P[Z] = s + u[(1 - q)^n - r^n]$$

If the output blocks also have the same probabilities as the input blocks, i.e., s = p, t = q, and u = r, then

$$P[Z''] = q + r[1 - (1 - q)^n]$$
$$P[Z'] = r^{n+1}$$
$$P[Z] = p + r[(1 - q)^n - r^n]$$

Similar expressions can be derived for other logic elements and they are summarized in Table II.

Table II Probability Equations for n-Input Logic Gates

Gate Type	$P[Z'']$	$P[Z']$	$P[Z]$
AND	$q + r[1-(1-q)^n] = q + E_n$	r^{n+1}	$p + r[(1-q)^n - r^n] = p + F_n$
OR	$q + r[(1-p)^n - r^n] = q + G_n$	r^{n+1}	$p + r[1-(1-p)^n] = p + H_n$
NAND	$q + r[(1-q)^n - r^n] = q + F_n$	r^{n+1}	$p + r[1-(1-q)^n] = p + E_n$
NOR	$q + r[1-(1-p)^n] = q + H_n$	r^{n+1}	$p + r[(1-p)^n - r^n] = p + G_n$
Where	$E_n = r[1 - (1 - q)^n]$ $F_n = r[(1 - q)^n - r^n]$ $G_n = r[(1 - p)^n - r^n]$ $H_n = r[1 - (1 - p)^n]$		

If p and/or q are very small, then higher order terms of p and/or q may be neglected to obtain the first order approximation. Table III shows probability expressions with this approximation.

VI. COMPARISON OF RELIABILITIES OF UNIFORM SECOND ORDER NETWORKS

A uniform second order network, as illustrated in Figure 6, is a two level network in which m first level gates feed one second level gate and each first level gate has exactly n inputs. Canonical realizations, universal logic modules, and multiplexers are all examples of uniform second order networks. TMR (triple modular redundancy) and some other networks may be considered as more complex networks composed of two or more uniform second order networks. It is therefore, interesting to compare the reliability of various uniform second order networks. Four types of uniform second order networks are

Table III Probability Expressions with First Order Approximation

	Gate Type	P [Z"]	P [Z']	P [Z]
p is small	AND	$q+r\left[1-(1-q)^n\right]$	r^{n+1}	$p+r\left[(1-q)^n-r^n\right]$
	OR	$q+r\left[1-np-r^n\right]$	r^{n+1}	$p+nrp$
	NAND	$q+r\left[(1-q)^n-r^n\right]$	r^{n+1}	$p+r\left[1-(1-q)^n\right]$
	NOR	$q+nrp$	r^{n+1}	$p+r\left[1-np-r^n\right]$
q is small	AND	$q+nrq$	r^{n+1}	$p+r\left[1-nq-r^n\right]$
	OR	$q+r\left[(1-p)^n-r^n\right]$	r^{n+1}	$p+r\left[1-(1-p)^n\right]$
	NAND	$q+r\left[1-nq-r^n\right]$	r^{n+1}	$p+nrq$
	NOR	$q+r\left[1-(1-p)^n\right]$	r^{n+1}	$p+r\left[(1-p)^n-r^n\right]$
Both p and q are small	AND	$q+nrq$	r^{n+1}	$p+r\left[1-nq-r^n\right]$
	OR	$q+r\left[1-np-r^n\right]$	r^{n+1}	$p+nrp$
	NAND	$q+r\left[1-nq-r^n\right]$	r^{n+1}	$p+nrq$
	NOR	$q+nrp$	r^{n+1}	$p+r\left[1-np-r^n\right]$

considered here. They are AND-OR, OR-AND, NAND-NAND and NOR-NOR. Assuming that the input and output circuits in all gates have the same probabilities, the probability expressions of these four types of networks are given in Table IV. The expressions may be simplified if approximations are used.

Table IV Probability Expressions of Uniform Second Order Networks

Network Type	P [Z"]	P [Z']	P [Z]
AND-OR	$q+r\left[(1-J_n)^m-r^{m(n+2)}\right]$	$r^{m(n+2)+1}$	$p+r\left[1-(1-J_n)^m\right]$
OR-AND	$q+r\left[1-(1-K_n)^m\right]$	$r^{m(n+2)+1}$	$p+r\left[(1-K_n)^m-r^{m(n+2)}\right]$
NAND-NAND	$q+r\left[(1-L_n)^m-r^{m(n+2)}\right]$	$r^{m(n+2)+1}$	$p+r\left[1-(1-L_n)^m\right]$
NOR-NOR	$q+r\left[1-(1-M_n)^m\right]$	$r^{m(n+2)+1}$	$p+r\left[(1-M_n)^m-r^{m(m+2)}\right]$

n = number of inputs to the first level gates
m = number of inputs to the second level gate

$$J_n = p + r\left\{p + r\left[(1-q)^n - r^n\right]\right\}$$

$$K_n = q + r\left\{q + r\left[(1-p)^n - r^n\right]\right\}$$

$$L_n = q + r\left\{q + r\left[(1-q)^n - r^n\right]\right\}$$

$$M_n = p + r\left\{p + r\left[(1-p)^n - r^n\right]\right\}$$

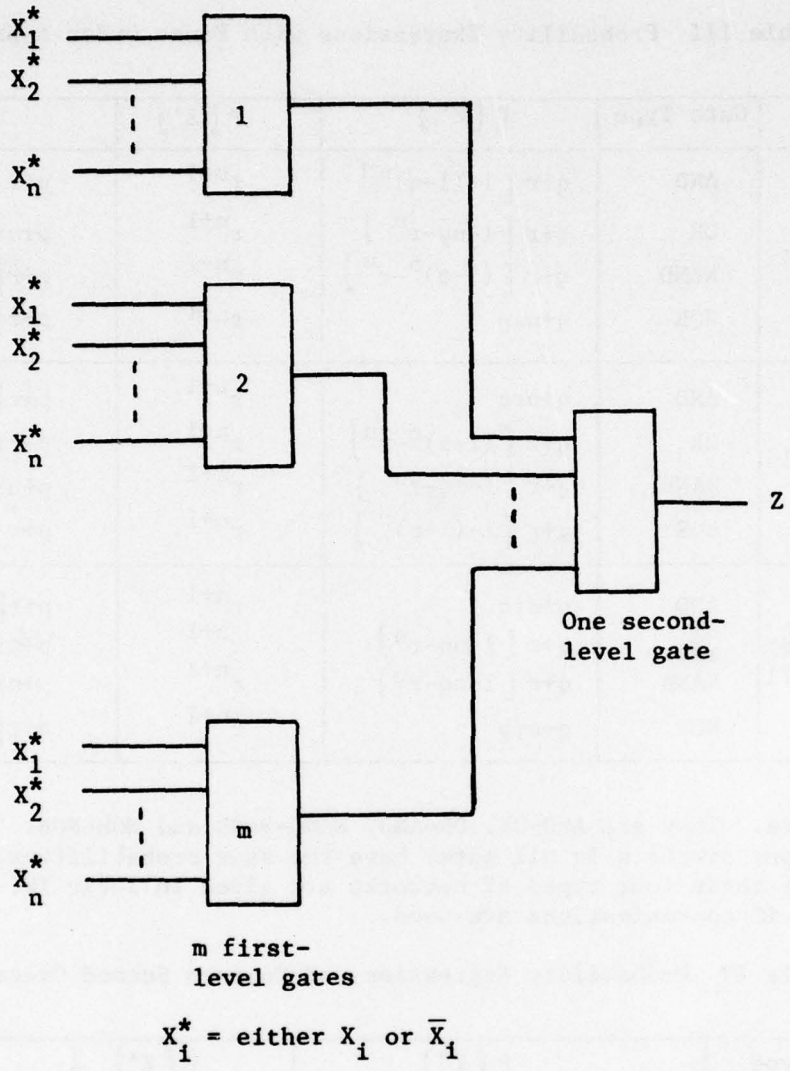


Figure 6. Uniform second order network.

Note from Table IV that

(1) When $p = q$, $J_n = K_n = L_n = M_n$.

(2) When $p > q$, $J_n > M_n$, $J_n > L_n$, $J_n > K_n$, $M_n > K_n$, and $L_n > K_n$ (J_n is the largest and K_n is the smallest).

(3) When $P < q$, $K_n > M_n$, $K_n > L_n$, $K_n > J_n$, $M_n > J_n$, and $K_n > J_n$ (K_n is the largest and J_n is the smallest).

Hence, the following general conclusions may be drawn with regard to these four types of networks.

(1) The overall network reliability, $P[Z] = \frac{1}{2}$, is the same for all four types of networks.
(2) If $p = q$, the AND-OR and NAND-NAND networks have identical probabilities; the OR-AND and NOR-NOR networks also have identical probabilities.
(3) If $p > q$, AND-OR and OR-AND networks have lower (higher) probability of committing logic-0 (logic-1) faults than NAND-NAND and NOR-NOR networks respectively.
(4) If $p < q$, the opposite of (3) is true.

VII. CONCLUSION

A thorough reliability analysis is necessary to define the design objectives of a product. The ternary algebraic approach presented in this paper not only allows engineers to compute simply the logic network reliability but also the probabilities of the network producing a logic-1 and a logic-0 fault. The approach is straightforward and can be easily followed by logic designers. An area for further research would be in the use of the algebraic (hence probablistic approach) for reliability design.

References

1. I-Ngo Chen, "Analysis and Reliability Estimation for Probablistic Switching Circuits", IEEE Trans. on Reliability Vol. R-20, pp. 36-38, Feb. 1971.
2. Eichelberger, E.B., "Hazard Detection in Combinational and Sequential Switching Circuits", IBM J. of Research and Development, vol. 9, pp. 90-99, March 1965.
3. M. Yoeli and S. Rinon, "Application of Ternary Algebra to the Study of Static Hazards", J. of ACM. Vol. 11, pp. 84-97, Jan. 1964.
4. Fantauzzi, G., "An Algebraic Model for the Analysis of Logical Circuits", IEEE Transactions on Computers, vol. C-23, pp. 576-581, June 1974.
5. Breuer, M.A., "A Note on Three-Valued Logic Simulation", IEEE Transactions on Computers, vol. C-21, pp. 299-402, April 1972.
6. E. L. Post, "Introduction to a General Theory of Elementary Propositions", American J. of Math. Vol. 43, pp. 163-185, 1921.
7. Papoulis, A., Probability, Random Variables, and Stochastic Process, New York; McGraw Hill, 1965.

TERNARY LOGIC SYSTEM BASED ON T-GATE

Tatsuo HIGUCHI and Michitaka KAMEYAMA
Department of Electronic Engineering, Faculty of Engineering,
Tohoku University, Aoba, Aramaki, Sendai, Japan

ABSTRACT

This paper presents that an arbitrary ternary logic system including combinational and sequential circuits can be realized with ternary T-gates which construct a complete system. The mathematical structure of the ternary T-gate is made clear which is of great use for the synthesis of the T-gate network. In particular the canonical form of the T-gate network is simple and expressed in a tree expansion. The implementation of the ternary T-gate is done by the electronic switching circuit. In the T-gate thus obtained the level loss is no practical limitation and many values of fan-out can be allowed. The combinational circuit of an arbitrary ternary switching function of n variables is systematically realized with $(3^n-1)/2$ T-gates in the tree expansion. For the purpose of realizing ternary sequential circuits, a tristable multivibrator and one-trit ternary shift resister can also be constructed by means of two and four T-gates, respectively. From this it is established that the T-gate as a basic building block can not be used as the swithing gate, but also the ternary storage element. As an example of the ternary logic system including both the combinational and sequential circuits, a serial 3-input ternary adder is built up by the T-gate network.

I. INTRODUCTION

It is expected that the ternary logic system consisting of only one type of gates would be realized. So far, a T-gate which constructs a complete system [1] has been implemented by the electronic circuit [2-6]. For the purpose of synthesizing the T-gate network, however, the mathematical and circuital properties of the T-gate have not yet been investigated in detail. Moreover, it has not been shown that the T-gate can not only be used as the switching gate, but also the ternary storage element.

In this paper the mathematical and circuital properties of the T-gate are studied. On the basis of the result thus obtained, the T-gate network [7] as the combinational circuit is systematically realized. Moreover, a tristable multivibrator and a ternary shift resister also are constructed by using the T-gates. Using the ternary shift resister, a maximum-length sequence generator is obtained. As an example of the ternary logic system including both the combinational and sequential circuits, a serial 3-input ternary adder is built up from the T-gates.

II. MATHEMATICAL PROPERTIES OF TERNARY T-GATE

The T-gate described here is essentially that of A. Church [8], A Rose [9], and C. Y. Lee and W. H. Chen [1]. We will now establish several more properties which the T-gate has. Let $L=\{1,0,-1\}$ be the set of logic values.

Definition 1. A ternary switching function $T(p,q,r;s)$ is defined [1] as

$$T(p,q,r;s) = p \times J_1(s) + q \times J_0(s) + r \times J_{-1}(s)$$

where the variables p, q, r and $s \in L=\{1,0,-1\}$,

and where the product $\times$, the sum +, and the J operation are defined as follows

$$J_k(s) = \begin{cases} 1 & \text{if } s=k \\ & , \quad x \times y = \min(x,y), \quad x + y = \max(x,y). \\ -1 & \text{if } s \neq k \end{cases}$$

Physically, a T-gate is a 4-input gate the output of which assumes the values of p, q or r according as the value of s is 1, 0 or -1. Thus s is called a control signal on the T-gate.

Theorem 1. $T(1,0,-1;f(X)) = f(X)$
$T(f(X),f(X),f(X);x_i) = f(X)$
where $X = (x_1,\ldots,x_i,\ldots,x_n) \in \{1,0,-1\}$ = vector on n ternary variables

$f(X)$= an arbitrary ternary switching function of n variables.

Proof. This is obvious according to the definition of the T-gate.
If a given ternary switching function is synthesized by the T-gate network, then the assignment of the control signal must be done so that minimization of the number of T-gates is realized. In this case Theorem 1 is very basic and important [10].

Theorem 2. $T(f_1(X),f_0(X),f_{-1}(X);x_i)$
$= T(f_1(x_1,\ldots,1,\ldots,x_n),f_0(x_1,\ldots,0,\ldots,x_n),f_{-1}(x_1,\ldots,-1,\ldots,x_n);x_i)$
where $f_1(X)$, $f_0(X)$ and $f_{-1}(X)$ are arbitrary ternary switching functions of n variables.

Proof. At the left-hand side of Theorem 2 we have

$f_1(x_1,\ldots,1,\ldots,x_n)$ if $x_i=1$
$f_0(x_1,\ldots,0,\ldots,x_n)$ if $x_i=0$
$f_{-1}(x_1,\ldots,-1,\ldots,x_n)$ if $x_i=-1$.

Hence the theorem is obvious.

Theorem 3. An arbitrary ternary switching function f(X) of n variables can completely be expressed based on an algebra using T-gates [1], and realized with $(3^n-1)/2$ T-gates at its maximum.

Proof. We shall proof the theorem by mathematical induction. When n=1, f(X) is a unary function f(x). Let p, q and r be constant functions, and s be x. Then, by Definition 1, f(X) can obviously be realized with one T-gate. Now suppose that the theorem is valid for n, and let us show that this implies its validity for n+1. By Theorems 1 and 2,

$$f(x_1,\ldots,x_{n+1}) = T(f(x_1,\ldots,x_{n+1}),f(x_1,\ldots,x_{n+1}),f(x_1,\ldots,x_{n+1});x_i)$$
$$= T(f(x_1,\ldots,1,\ldots,x_{n+1}),f(x_1,\ldots,0,\ldots,x_{n+1}),$$
$$f(x_1,\ldots,-1,\ldots,x_{n+1});x_i).$$

At the right-hand side of the above equation, each function f( ) is the ternary switching function of n variables. Therefore, by assumption, $f(x_1,\ldots,x_{n+1})$ can be realized with $\{(3^n-1)/2\}\times 3 + 1 = (3^{n+1}-1)/2$ T-gates at its maximum. If the minimization of a given switching function is carried out, then the number of T-gates can be reduced.

Theorem 4. An arbitrary ternary switching function f(X) of n variables can be expanded
$f(X) = T(f_1(X'),f_0(X'),f_{-1}(X');x_i)$
where $X' \cap x_i = 0$ (empty set), $X' \cup x_i = X$,
and where $f_1(X')=f(x_1,\ldots,1,\ldots,x_n)$, $f_0(X')=f(x_1,\ldots,0,\ldots,x_n)$, and

$f_{-1}(X')=f(x_1,\ldots,-1,\ldots,x_n)$.

Proof. By Theorems 1 and 2,

$$f(X) = T(f(X),f(X),f(X);x_i)$$
$$= T(f(x_1,\ldots,1,\ldots,x_n),f(x_1,\ldots,0,\ldots,x_n),f(x_1,\ldots,-1,\ldots,x_n);x_i)$$
$$= T(f_1(X'),f_o(X'),f_{-1}(X');x_i).$$

Example. The ternary switching function of two variables $f(x_1,x_2)$ given by the truth table of Table 1 can be realized by four T-gates as follows

$$f(x_1,x_2) = T(f(1,x_2),f(0,x_2),f(-1,x_2);x_1)$$
$$= T(T(0,1,-1;x_2),T(-1,1,0;x_2),T(1,0,1;x_2);x_1).$$

This T-gate network can be expressed in the tree expansion of Fig. 1. Each node corresponds to the T-gate. In general we can systematically realize a given function of n variables in successive levels $L_1, L_2,\ldots,L_n$ of T-gates as shown in Fig. 2.

Theorem 5. In the tree expansion of T-gates the level L_n consists of 3^{n-1} T-gates at its maximum.

Proof. By Theorem 3 the level L_n consists of $(3^n-1)/2 - (3^{n-1}-1)/2 = 3^{n-1}$ T-gates at its maximum.

Definition 2. The dual of x_i can be defined [1] by

$\overline{x}_i = T(-1,0,1;x_i)$. Obviously $\overline{\overline{x}}_i = x_i$.

Theorem 6. $\overline{T(f_1(X'),f_o(X'),f_{-1}(X');x_i)} = T(\overline{f_1(X')},\overline{f_o(X')},\overline{f_{-1}(X')};x_i)$.

Proof. The left-hand side of the above equation yields

$\overline{f_1(X')}$ if $x_i=1$

$\overline{f_o(X')}$ if $x_i=0$

$\overline{f_{-1}(X')}$ if $x_i=-1$.

Hence the theorem is obvious.

Theorem 7. Let $g\{x\}$ be an arbitrary unary switching function, and let $f(X) = T(f_1(X'),f_o(X'),f_{-1}(X');x_i)$, then

$$g\{f(X)\} = T(g\{f_1(X')\},g\{f_o(X')\},g\{f_{-1}(X')\};x_i).$$

Proof. The left-hand side can be expressed as

$g\{f_1(X')\}$ if $x_i=1$

$g\{f_o(X')\}$ if $x_i=0$

$g\{f_{-1}(X')\}$ if $x_i=-1$.

The right-hand side can also be expressed in common with the left-hand side. Hence the right-hand and left-hand sides are identical.

Theorem 8. $T(f_1(X'),f_o(X'),f_{-1}(X');g\{x\}) = T(f_{g\{1\}}(X'),f_{g\{0\}}(X'),f_{g\{-1\}}(X');x)$.

Proof. The left-hand side of the above equation becomes $f_{g\{x\}}(X')$, since

$f_1(X')$ if $g\{x\}=1$

$f_o(X')$ if $g\{x\}=0$

x_1 \ x_2	1	0	-1
1	0	1	-1
0	-1	1	0
-1	1	0	1

Table 1

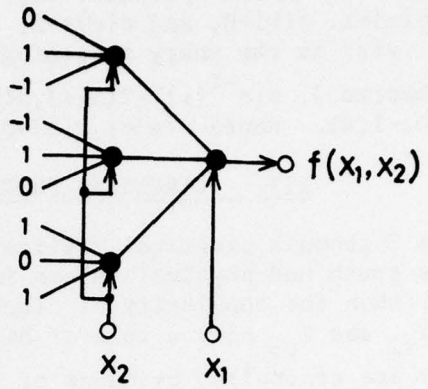


Fig. 1

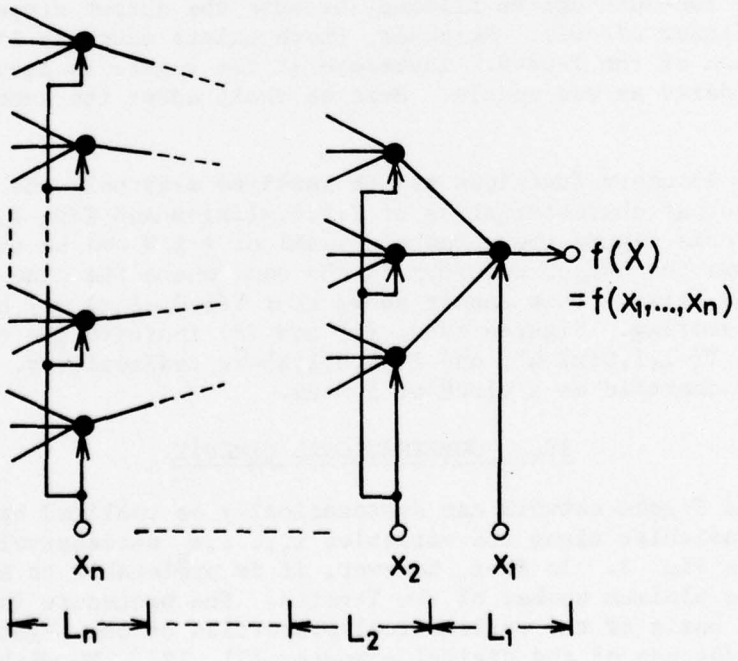


Fig. 2

$f_{-1}(X')$ if $g\{x\}=-1$.

The right-hand side can also be expressed as $f_{g\{x\}}(X')$. Hence the theorem is obvious.

Theorem 9. Let $\sigma\{x\}=T(\sigma\{1\},\sigma\{0\},\sigma\{-1\};x)$ be an arbitrary permutation, and let $\sigma^{-1}\{x\}=T(a,b,c;x)$ be an inverse permutation, where a, b and c are constant functions. Then we obtain $\sigma\{a\}=1$, $\sigma\{b\}=0$, and $\sigma\{c\}=-1$, so that a, b and c are uniquely defined.

Proof. $\sigma\{x\}$ is the unary switching function and contained in the set of $g\{x\}$. Thus, by Theorem 7, $\sigma\{\sigma^{-1}\{x\}\}=T(\sigma\{a\},\sigma\{b\},\sigma\{c\};x)$. Obviously $\sigma\{\sigma^{-1}\{x\}\}=x$. By Theorem 1, $x=T(1,0,-1;x)$. Hence $\sigma\{a\}=1,\sigma\{b\}=0$, and $\sigma\{c\}=-1$.

III. CIRCUITAL PROPERTIES OF TERNARY T-GATE

Figure 3 shows a practical realization of the T-gate [4], where the relation between the truth and physical values is defined by Table 2. If this T-gate can be integrated, then the complexity of circuit will be not an important subject. Transistors T_{r1}, T_{r2} and T_{r3} play a role of bilateral switches. The base voltages of these transistors are controlled by means of transistors $T_{r4},\ldots,T_{r8}$. The complementary transistors T_{r9} and T_{r10} form the emitter-follower circuit of which the level shift is compensated by insertion of diodes D_1 and D_2.

In the T-gate the threshold-level of the input terminal s is set at $\pm$ 1 V. This value can be adjusted by the value of $R_1,\ldots,R_5$. The level loss is no practical limitation and many fan-outs can be allowed, because the output stage is constructed by the emitter-follower circuit. Moreover, there exists scarcely limitation for the mutual connection of the T-gate. Therefore if the T-gate is actually integrated, then it can be considered as one module. Here we shall adopt the symbol of Fig. 4 for a T-gate [2].

All of the 27 unary functions can be realized with only one T-gate. As examples, the input and output characteristics of $T(1,0,-1;x)=x$ and $T(0,-1,1;x)=x''$ are presented in Fig. 5. In this figure the threshold-level of $\pm$ 1 V can be observed at the input x. Figure 6(a) shows the output waveform in the case where the sine-wave input as x is applied to $T(1,0,-1;x)$. This result shows that $T(1,0,-1;x)$ may be used for the ternary signal level-restoring. Figures 6(b), (c) and (d) indicate the output waveforms of $T(0,-1,1;x)=x''$, $T(-1,1,0;x)=x'$, and $T(-1,0,1;x)=\bar{x}$, respectively. At present the T-gate is sufficiently operated at a clock of 1 M Hz.

IV. COMBINATIONAL CIRCUIT

The general T-gate network can systematically be realized by expanding a given function of n variables along the variables $x_1,\ldots,x_n$ successively in the form of a tree as shown in Fig. 2. In fact, however, it is preferable to synthesize the T-gate network with the minimum number of the T-gates. The procedure for the minimization obtained on the basis of the mathematical properties of the T-gate is practically carried out by the use of the digital computer [11, 12]. We wish to present this matter on another occasion.

As an example of the combinational switching circuit, the ternary half adder for signed ternary system is built up by the T-gate network. The sum S(a,b) and the carry

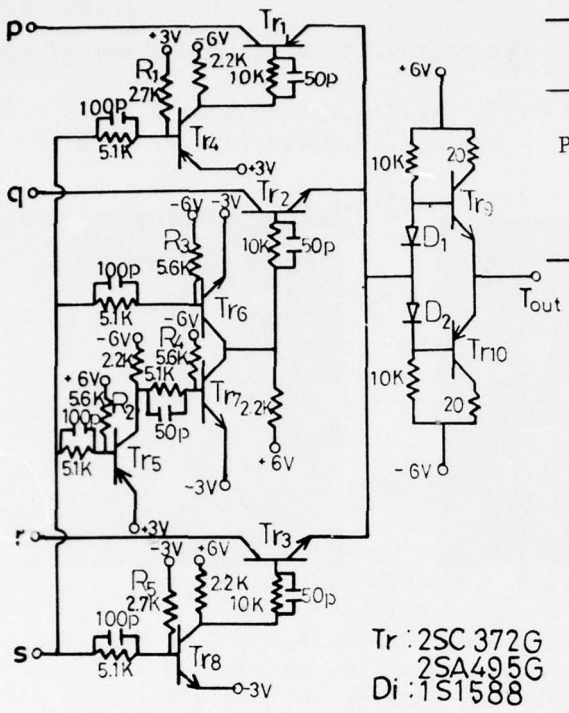


Fig. 3

Truth value	1	0	-1
Physical value	+3V	0V	-3V

Table 2

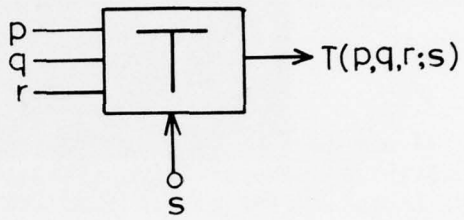
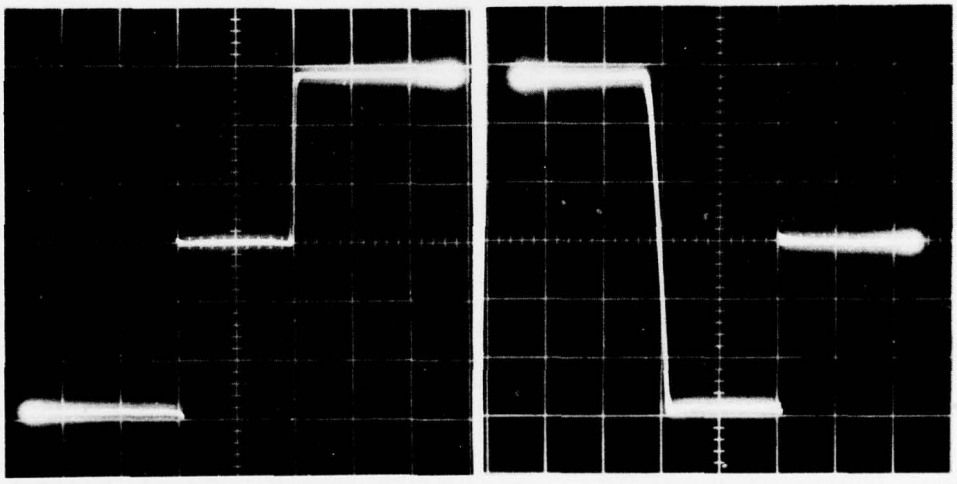


Fig. 4

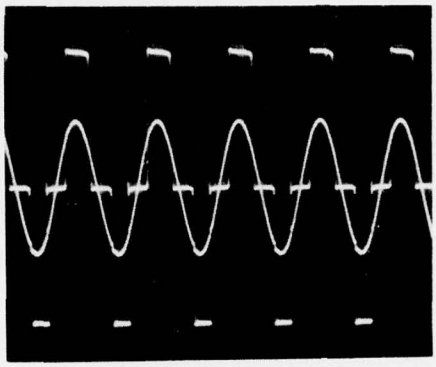


(a) T(1,0,-1;x)

(b) T(0,-1,1;x)

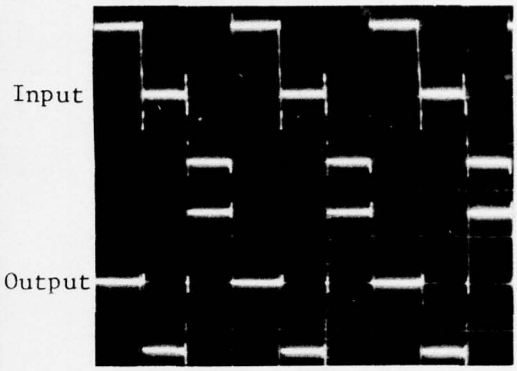
Horizontal : x 1V/div.
Vertical : T_{out} 1V/div.

Fig. 5



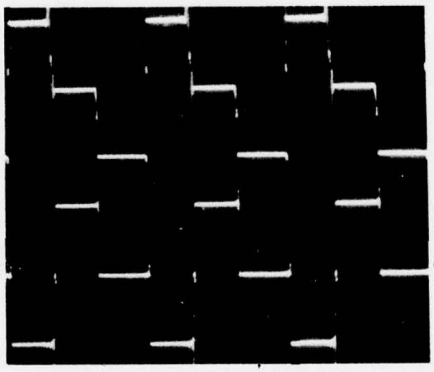
5μs/div., 1V/div.

(a) T(1,0,-1;x)



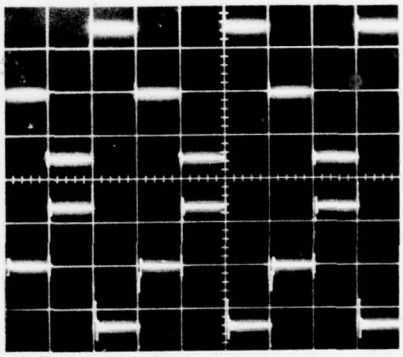
5μs/div., 2V/div.

(b) T(0,-1,1;x)



5μs/div., 2V/div.

(c) T(-1,1,0;x)



5μs/div., 2V/div.

(d) T(-1,0,1;x)

Fig. 6

C(a,b) given by the truth table of Table 3 are expressed by using Theorems 1 and 4 as

$$S(a,b) = T(S(a,1),S(a,0),S(a,-1);b) = T(T(-1,1,0;a),T(1,0,-1;a),T(0,-1,1;a);b)$$
$$= T(T(-1,1,0;a),a,T(0,-1,1;a);b) \qquad (1)$$

$$C(a,b) = T(C(a.1),C(a,0),C(a,-1);b) = T(T(1,0,0;a),T(0,0,0;a),T(0,0,-1;a);b)$$
$$= T(T(1,0,0;a),0,T(0,0,-1;a);b). \qquad (2)$$

In the above equation the minimization procedure is simply carried out without the use of the digital computer, because a given function has only two variables. From Eqs. (1) and (2) it follows that the half adder can be built up from six T-gates as shown in Fig. 7. Figure 8 shows the experimental result of the input and output waveforms of the half adder. It is found that the result is corresponding to the truth table of Table 3.

V. SEQUENTIAL CIRCUIT

In this section it is shown that the T-gate can not only be used as the switching gate, but also the ternary storage element. A tristable multivibrator is constructed by using the T-gate as a basic building block. Tristable states x=1, 0 and -1 can directly be obtained by T(1,0,-1;x)=x. In a single T-gate, however, the transition of state from one state to another can not be attained. The tristable multivibrator is realized with cross-coupling two T-gates as shown in Fig. 9. This multivibrator has the function analogous to the binary D flip-flop. In this figure the input signal goes through after one clock pulse of "1" is first applied. At the moment the clock pulse has returned to "0", the input signal is stored in the tristable multivibrator as it is because of feeding the output Y back into the terminal q.

Furthermore, a ternary shift resister which is basically a chain of master-slave tristable multivibrators can be constructed. Four T-gates are used to construct one-trit ternary shift resister as shown in Fig. 10. Figure 11 presents the time chart of the shift resister. In this figure the first and second stages are tristable multivibrators of which transitions of state are done at the front and back edges of a clock pulse C.P., respectively : At the first stage the input signal is set up when C.P.1="1", and then stored when C.P.1="0". In this case the first stage is separated from the second stage. Next, the signal stored in the first stage is transferred to the second stage when C.P.2="-1", and then stored in when C.P.2="0". In this case the input terminal of the first stage remains OFF.

Consider a maximum-length sequence generator [13] as an application of the shift resister. The generator obtained here is represented in block form as in Fig. 12. Ternary switching function $f(x_2,x_3)$ given by the truth table of Table 4 can be expressed as

$$f(x_2,x_3) = T(f(x_2,1),f(x_2,0),f(x_2,-1);x_3)$$
$$= T(T(0,-1,1;x_2),T(1,0,-1;x_2),T(-1,1,0;x_2);x_3)$$
$$= T(T(0,-1,1;x_2),x_2,T(-1,1,0;x_2);x_3). \qquad (3)$$

Figure 13 shows the output waveform of the maximum-length sequence generator with 3-level sequence of length 26 in Table 5.

S(a,b)

b \ a	1	0	-1
1	-1	1	0
0	1	0	-1
-1	0	-1	1

C(a,b)

b \ a	1	0	-1
1	1	0	0
0	0	0	0
-1	0	0	-1

Table 3

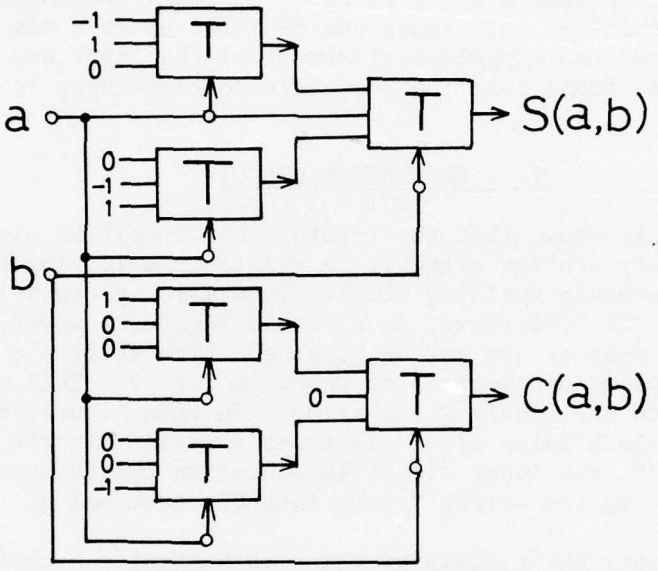
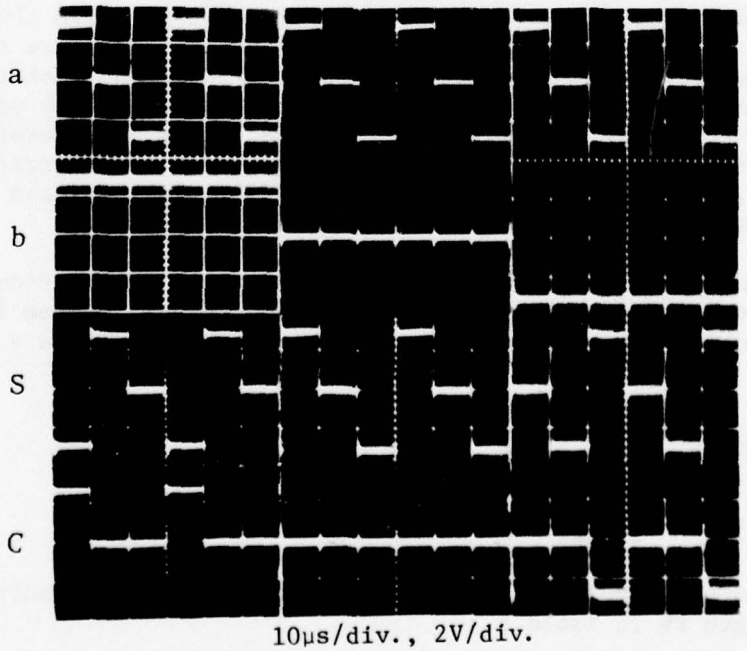


Fig. 7



10μs/div., 2V/div.

Fig. 8

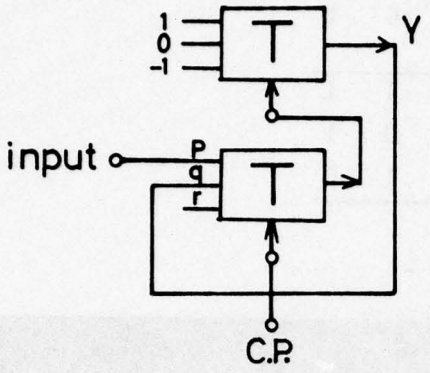


Fig. 9

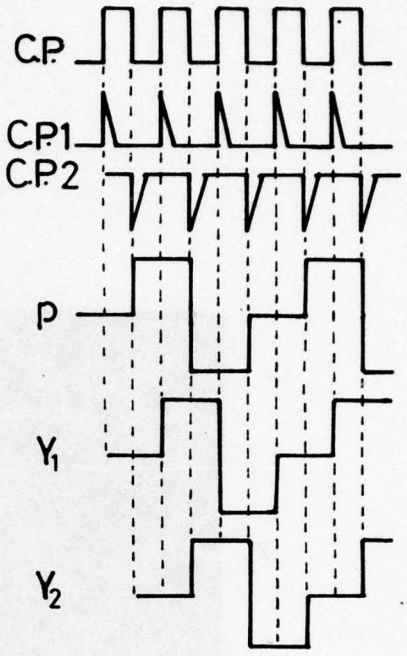


Fig. 11

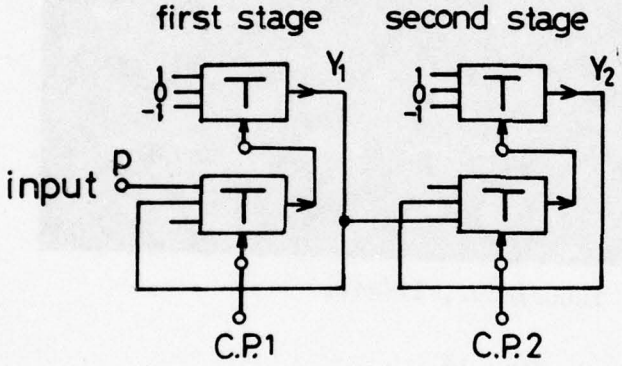


Fig. 10

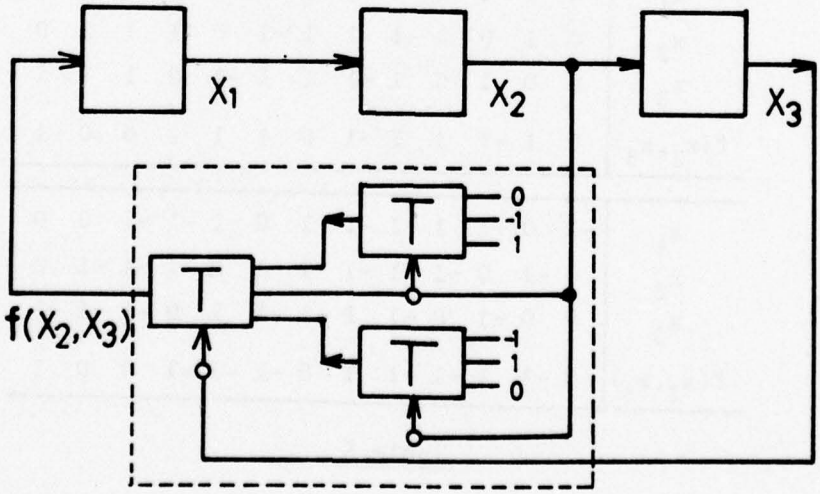
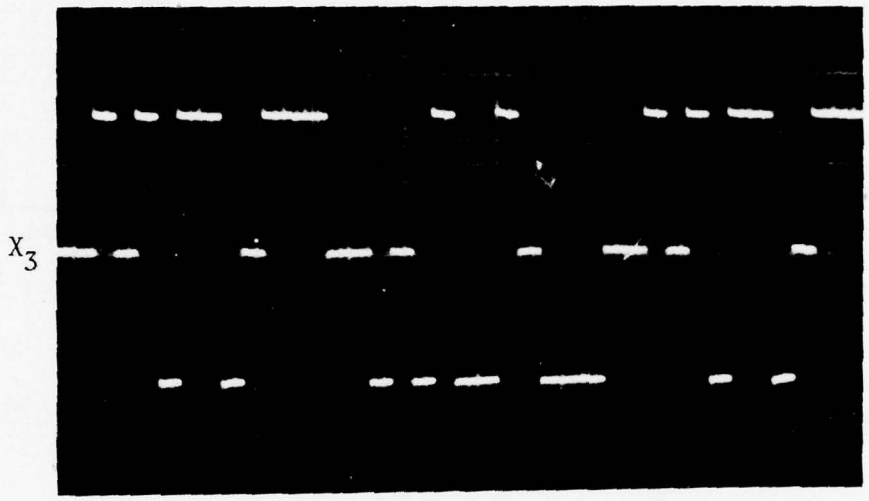


Fig. 12

X_3 \ X_2	1	0	-1
1	0	-1	1
0	1	0	-1
-1	-1	1	0

Table 4



100μs/div., 2V/div.

Fig. 13

x_1	1	0	1	-1	1	1	-1	0	1	1	1	0	0
x_2	0	1	0	1	-1	1	1	-1	0	1	1	1	0
x_3	0	0	1	0	1	-1	1	1	-1	0	1	1	1
$f(x_2,x_3)$	0	1	-1	1	1	-1	0	1	1	1	0	0	-1

x_1	-1	0	-1	1	-1	-1	1	0	-1	-1	-1	0	0
x_2	0	-1	0	-1	1	-1	-1	1	0	-1	-1	-1	0
x_3	0	0	-1	0	-1	1	-1	-1	1	0	-1	-1	-1
$f(x_2,x_3)$	0	-1	1	-1	-1	1	0	-1	-1	-1	0	0	1

Table 5

VI. SERIAL TERNARY ADDER

A ternary logic system including both the combinational and sequential circuits can be realized on the basis of the T-gate network. As an example of this sort of network, a serial 3-input ternary adder is built up. Table 6 indicates the truth table of the full adder. According to the canonical expansion of Theorem 3, the sum S(a,b,c) and the carry C(a,b,c) can be realized with 26 T-gates, since they are the ternary switching functions of three variables. From the minimization procedure applied, however, it follows that S(a,b,c) and C(a,b,c) can be expressed by 14 T-gates as follows

$$S(a,b,c) = T(T(0,-1,1;a),T(-1,1,0;a),a;\alpha) \tag{4}$$

$$\alpha = T(b,T(0,-1,1;b),T(-1,1,0;b);c) \tag{5}$$

$$C(a,b,c) = T(T(T(1,1,0;a),T(1,0,0;a),0;b),T(T(1,0,0;a),0,T(0,0,-1;a);b), T(0,T(0,0,-1;a),T(0,-1,-1;a);b);c). \tag{6}$$

The full adder is first constructed according to Eqs. (4), (5) and (6). Then the serial ternary adder can be obtained by feeding the output of carry back by the use of the one-trit shift resister as shown in Fig. 14.

Let A, B and C correspond to the internal states of the serial adder, i.e., the states of carry 1, 0 and -1. In this case the transition diagram is presented in Fig. 15. Let a,b/S denote the output of sum S in response to the inputs a and b. Thus, for example, consider periodic input pulses a=-1,0,1,..., and b=1,1,1,...,. We begin with the initial state A. Then the transition to the state B is done, since S=1 in response to a=-1 and b=1. Next, S=1 in response to a=0 and b=1, so that the state B is still held. If S=-1 in response to a=1 and b=1, then the transition from B to A is carried out. After this the cycle will be repeated.

Table 7 presents several examples of the output in response to the periodic input sequence. Figure 16 shows the input and output waveforms corresponding to Table 7. From this it is found that the serial ternary adder thus obtained is satisfactorily operated.

VII. CONCLUSION

This study has shown that a ternary T-gate as a basic building block is very useful because both the combinational and sequential circuits can be constructed by using the T-gates.

The T-gate can be considered to be a sort of universal logic module presented by S. S. Yau and C. K. Tang [14]. The T-gate network will be important with the potentialities of the integrated circuit. For the synthesis of the T-gate network, the mathematical and circuital properties of the T-gate have been studied.

ACKNOWLEDGMENT

The authors wish to thank Prof. T. Anayama for his guidance and encouragement.

S(a,b,c)

b,c \ a	1	0	-1
1 1	0	-1	1
1 0	-1	1	0
1 -1	1	0	-1
0 1	-1	1	0
0 0	1	0	-1
0 -1	0	-1	1
-1 1	1	0	-1
-1 0	0	-1	1
-1 -1	-1	1	0

C(a,b,c)

b,c \ a	1	0	-1
1 1	1	1	0
1 0	1	0	0
1 -1	0	0	0
0 1	1	0	0
0 0	0	0	0
0 -1	0	0	-1
-1 1	0	0	0
-1 0	0	0	-1
-1 -1	0	-1	-1

Table 6

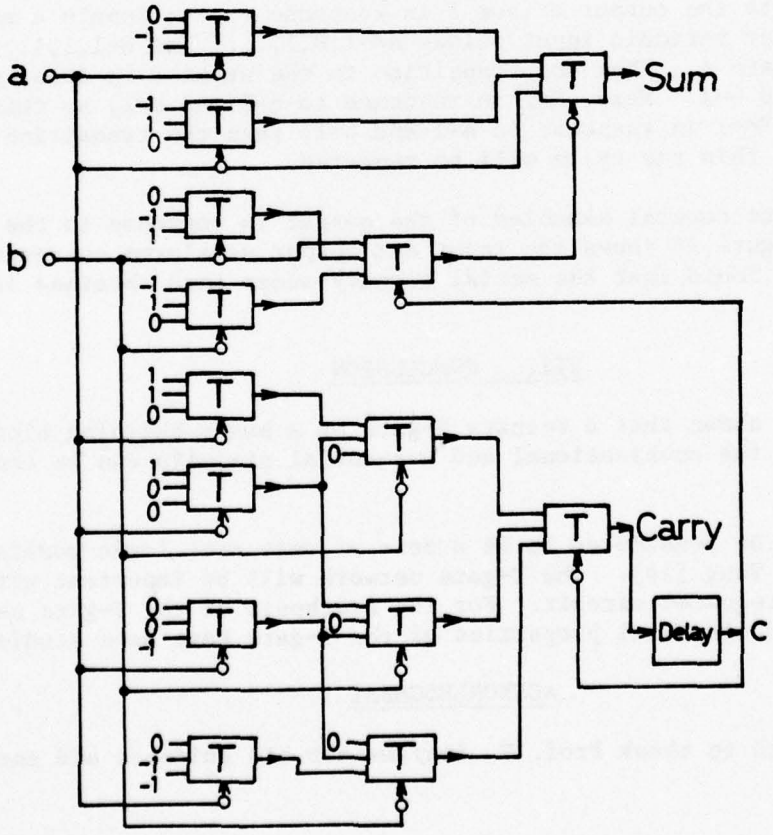


Fig. 14

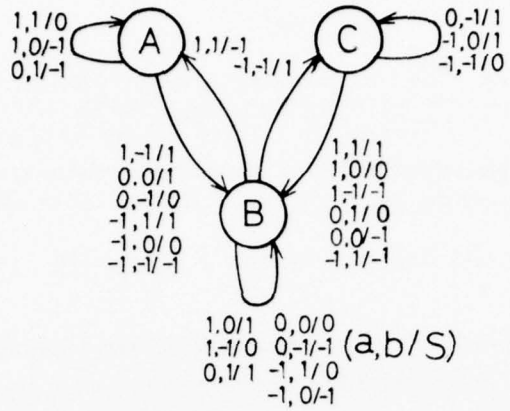
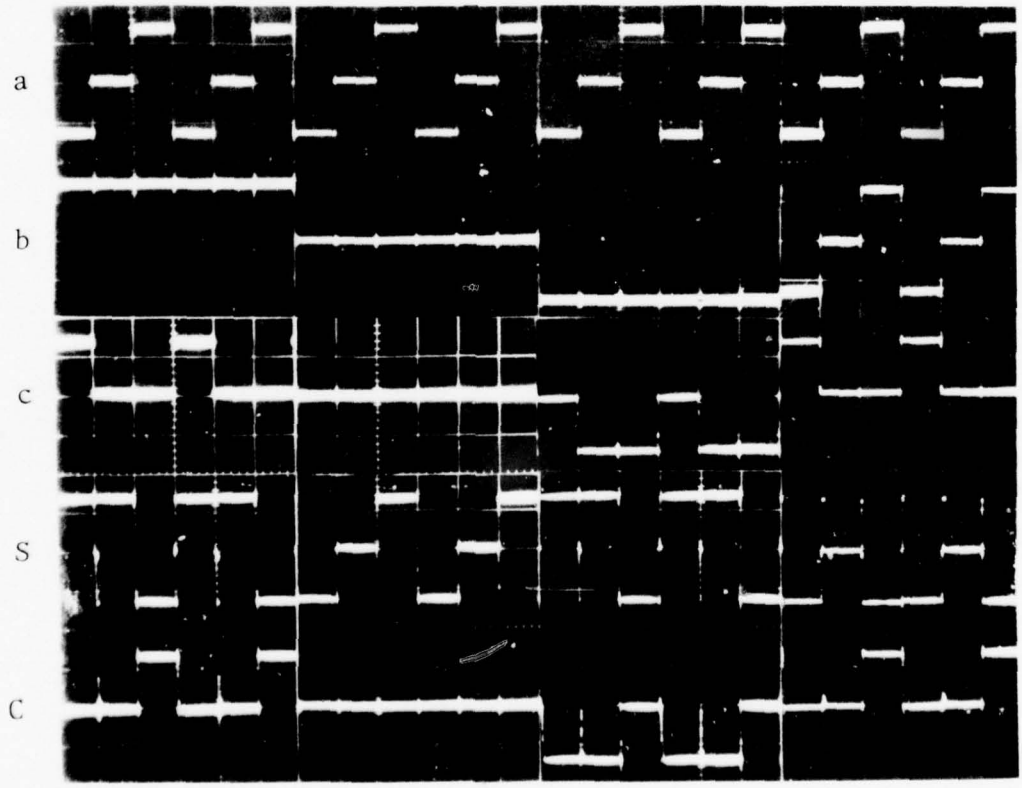


Fig. 15

a	-1	0	1	-1	0	1	-1	0	1	-1	0	1
b	1	1	1	0	0	0	-1	-1	-1	-1	0	1
c	1	0	0	0	0	0	0	-1	-1	1	0	0
C	0	0	1	0	0	0	-1	-1	0	0	0	1
S	1	1	-1	-1	0	1	1	1	-1	-1	0	-1

Table 7



50μs/div., 2V/div.

Fig. 16

REFERENCES

[1] C. Y. Lee and W. H. Chen, "Several-Valued Combinational Switching Circuit", AIEE Trans., vol. 75, pt. I, pp. 278-283, July 1956.

[2] S. Thelliez, "Note on the Synthesis of Ternary Combinational Networks Using T Gate Operators", Electronics Letters, vol. 3, pp. 204-205, May 1967.

[3] D. I. Porat, "Three-Valued Digital System", Proc. IEE, vol. 116, pp. 947-954, June 1969.

[4] T. Higuchi and M. Kameyama, "Ternary Logic Circuits Using T-Gates", IECE Japan Trans., vol. 56-D, pp. 270-271, April 1973.

[5] H. T. Mouftah and I. B. Jordan, "Integrated Circuits for Ternary Logic", Proc. 1974 International Symposium on Multiple-Valued Logic, pp. 285-302, May 1974.

[6] D. Etiemble and M. Israel, "A New Concept for Ternary Logic Elements", Proc. 1974 International Symposium on Multiple-Valued Logic, pp. 437-456, May 1974.

[7] S. Thelliez, Introduction à l'Étude des Structures Ternaires de Commutation, Gordon & Breach 1973 Paris.

[8] A. Church, "Conditioned Disjunction as a Primitive Connective for the Propositional Calculus", Portugaliae Mathematica, vol. 7, pp. 87-90, 1948.

[9] A. Rose, "Conditioned Disjunction as a Primitive Connective for the m-Valued Propositional Calculus", Mathematishe Annalen, vol. 123, pp. 76-78, 1951.

[10] T. Higuchi and M. Kameyama, "Synthesis of Multiple-Valued Logic Networks Based on Tree-Type Universal Logic Modules", (this issue).

[11] M. Kameyama and T. Higuchi, "Theoretical Study on the Synthesis of T-Gate Network", 1974 Convension Record of the IECE Japan, Aug. 1974.

[12] A. Kanomata and T. Higuchi, "Synthesis of T-Gate Network by Means of Digital Computer", 1974 Convension Record of the IECE Japan, Aug. 1974.

[13] H. Mine, T. Hasegawa, Y. Koga, M. Ikeda and T. Shintani, "Construction and Analysis of a Tri-Stable Circuit and Its Application to Ternary Feedback Shift Resister", IECE Japan Trans., vol. 52-C, pp. 443-450, Aug. 1969.

[14] S. S. Yau and C. K. Tang, "Universal Logic Modules and Their Applications", IEEE Trans. on Computers, vol. C-19, pp. 141-149, Feb. 1970.

BILINEAL SEPARABILITY OF TERNARY FUNCTIONS

Jorge Nazarala[+] Claudio Moraga[++]
Universidad de Chile Universität Dortmund
Chile West Germany

Abstract: A new type of Separability of Ternary Functions, namely Bilineal, is defined. Necessary and sufficient conditions for a function to have Bilineal Separability are given, and test parameters are tabulated. It is shown that use of Bilineal Separability allows simple realization of many functions which are not linearly separable, as well as simpler realization of threshold functions of many variables.

Index Terms: Ternary Threshold Logic, Threshold synthesis, Bilineal Separability.

1. Introduction:

It has been shown that there are 471 2-place and 85,629 3-place ternary threshold functions (1,2). These figures show first, that threshold functions are only a low fraction of the corresponding set of n-place ternary functions and then, that the yet unknown absolute number of 4-place ternary threshold functions must be very large.

This suggests the convenience of looking for other kind of separability to realize a larger number of functions and also, to allow simplified test and realization of 4-place functions.

[+] Dept. E.E.,Universidad de Chile, Santiago, Chile.

[++] Alexander von Humboldt Research Fellow, Abteilung Informatik, Universität Dortmund, Bundesrepublick Deutschland, on leave from: Dept. of Computer Science, Universidad Santa Maria, Valparaiso, Chile.

This paper shows that deleting the paralellismus condition of the separating planes of the classical threshold function equation, thus obtaining what here is called Bilineal Separation, the above mentioned goals may be achieved.

2. Notation, definition and theorems:

Let $V=(-1,0,1)$ be the set of possible values for a ternary variable. Let the space vector V^n be the set of vectors $\underline{v}_i=(v_{i1},v_{i2},\ldots,v_{in})$, $(i=1,2,\ldots,3^n)$ which represent all the possible state assignments to the variables $X_1,X_2,\ldots,X_n$. A ternary function $F(\underline{X})=F(X_1,X_2,\ldots,X_n)$ sets a partition over V^n defining three blocks F_-, F_0, and F_+ which are mapped onto -1,0 and 1 respectively.

Def. 1: $F(\underline{X})$ is strictly bilineal separable (SBS), iff $\exists$ $\underline{w}_1$ and $\underline{w}_2 \in Z^n$ and two real thresholds T_1 and T_2 such that:

$$\begin{aligned} F(\underline{X})=1 &\iff T_1 < \underline{w}_1\cdot\underline{X} \quad T_2 < \underline{w}_2\cdot\underline{X} \\ F(\underline{X})=0 &\iff \underline{w}_1\cdot\underline{X} < T_1 \quad T_2 < \underline{w}_2\cdot\underline{X} \qquad (1) \\ F(\underline{X})=-1 &\iff \underline{w}_1\cdot\underline{X} < T_1 \quad \underline{w}_2\cdot\underline{X} < T_2 \end{aligned}$$

i.e.: $F(\underline{X})$ is SBS iff there exist a set of two not necessarily parallel hyperplanes in $\mathbb{R}^n$, which separate F_+ from F_0 and F_0 from F_- respectively.

Obviously, every threshold function is also SBS.

Theorem 1: $F(\underline{X})$ is SBS iff $\exists!$ two two-valued Basis-functions $F_1(\underline{X})$ and $F_2(\underline{X})\in(-1,1)$ such that:

$$(F_2)_- \subset (F_1)_- \qquad (2)$$

$$\frac{F_1(\underline{X}) + F_2(\underline{X})}{2} = F(\underline{X}) \qquad (3)$$

Proof:

=> : Let $F(\underline{X})$ be SBS

$$\text{define } F_1(\underline{X}) = 1 \iff F(\underline{X}) = 1$$

$$-1 \iff F(\underline{X}) \in (-1,0) \qquad (4)$$

$$F_2(\underline{X}) = 1 \iff F(\underline{X}) \in (0,1)$$

$$-1 \iff F(\underline{X}) = -1 \qquad (5)$$

then: $(F_2)_- \subset (F_1)_-$

and $$\frac{F_1(\underline{X}) + F_2(\underline{X})}{2} = F(\underline{X})$$

It is easy to see from eqs. 1, 4 and 5, that both Basis-functions are two-valued threshold functions:

$$\text{i.e.: } F_1(\underline{X}) : (\underline{w}_1, T_1) \text{ , } F_2(\underline{X}) : (\underline{w}_2, T_2) \qquad (6)$$

From eqs. 1,2 and 3, Basis-functions are unique:

$$F(\underline{X}) = 1 \quad => \quad F_1(\underline{X}) = F_2(\underline{X}) = 1$$

$$F(\underline{X}) = -1 \quad => \quad F_1(\underline{X}) = F_2(\underline{X}) = -1 \qquad (7)$$

$$F(\underline{X}) = 0 \quad => \quad F_1(\underline{X}) = -1 \wedge F_2(\underline{X}) = 1$$

<= : Let $F_1(\underline{X})$ and $F_2(\underline{X})$ be as defined by the theorem. Then $\exists$ $\underline{w}_1$, $\underline{w}_2$, T_1 and T_2 which realize these functions and satisfy eq. 1. Thence, $F(\underline{X})$ is SBS.

According to theorem 1, a general realization for a SBS function is possible, as shown in fig. 1.

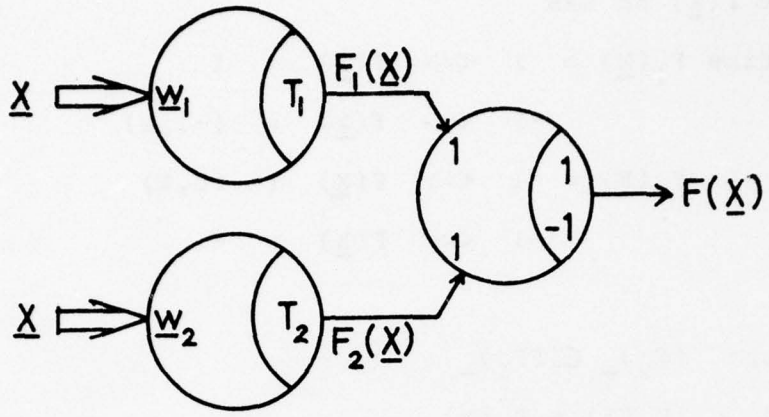


Fig. 1: General realization of a SBS function.

Theorem 2: (3)

Let $F(\underline{X})$ be a SBS function, with $F_1(\underline{X}):(\underline{w}_1,T_1)$ and $F_2(\underline{X}):(\underline{w}_2,T_2)$ as Basis-functions. Then $F(\underline{X})$ may be realized with only two threshold cascaded gates, as shown in fig. 2, when either of two sets of conditions is satisfied:

i) $n\,T_2 > 0$

$k=\max\,(\underline{w}_2.\underline{X} \mid F(\underline{X}) = 0)$

$k < (2n + 1)\,T_2$ (8)

$(k - nT_2) < H < (n + 1)T_2$

$L = (1-n)T_2$

ii) $n\,T_1 > 0$

$k = \min\,(\underline{w}_1.\underline{X} \mid F(\underline{X}) = 0)$

$k > (1-2n)\,T_1$

$H = (n + 1)\,T_1$ (9)

$(1 - n)\,T_1 < L < (k + nT_1)$

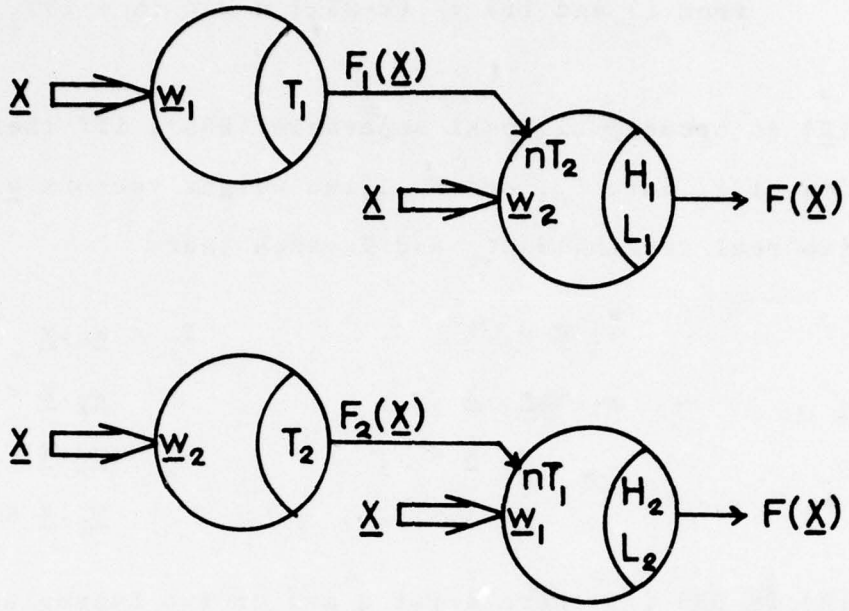


Fig. 2: Cascaded Threshold realization.

Proof: (first set).

let $\underline{w} = (nT_2, \underline{w}_2)$, $\underline{X}' = (\underline{X} , F_1(\underline{X}))$

i) $F(\underline{X}) = 1 \Rightarrow T_2 < \underline{w}_2 . \underline{X} \quad F_1(\underline{X}) = 1$

$\quad$ i.e.: $(n + 1) T_2 < \underline{w} . \underline{X}'$

ii) $F(\underline{X}) = 0 \Rightarrow T_2 < \underline{w}_2 . \underline{X} \leqslant k \quad F_1(\underline{X}) = -1$

$\quad$ with $k = \max (\underline{w}_2 . \underline{X} \mid F(\underline{X}) = 0)$

$\quad$ i.e.: $(1-n) T_2 < \underline{w} . \underline{X}' \leqslant (k - nT_2)$

iii) $F(\underline{X}) = -1 \Rightarrow \underline{w}_2 . \underline{X} < T_2 \quad F_1(\underline{X}) = -1$

$\quad$ i.e.: $\underline{w} . \underline{X}' < (1-n) T_2$

thus: $(k - nT_2) < (n + 1) T_2$

$\quad (1-n) T_2 < (n + 1) T_2$

then: $k < (2n+1)T_2$ and $nT_2 > 0$

from ii) and iii): $L = (1-n)T_2$

from i) and ii): $(k-nT_2) < H < (n+1)T_2$

Def 3: $F(\underline{X})$ is broadly bilineal separable (BBS), iff there exists a partition of F_0 into F_{01} and F_{02}, two weight vectors $\underline{w}_1$ and $\underline{w}_2 \varepsilon \mathbb{Z}^n$ and two real thresholds T_1 and T_2 such that:

$$
\begin{array}{lll}
\forall\, \underline{X} \varepsilon F_+ & T_1 < \underline{w}_1 \cdot \underline{X} & T_2 < \underline{W}_2 \cdot \underline{X} \\
\forall\, \underline{X} \varepsilon F_{01} & T_1 < \underline{w}_1 \cdot \underline{X} & \underline{w}_2 \cdot \underline{X} < T_2 \\
\forall\, \underline{X} \varepsilon F_{02} & \underline{w}_1 \cdot \underline{X} < T_1 & T_2 < \underline{w}_2 \cdot \underline{X} \\
\forall\, \underline{X} \varepsilon F_- & \underline{w}_1 \cdot \underline{X} < T_1 & \underline{w}_2 \cdot \underline{X} < T_2
\end{array}
\qquad (1o)
$$

i.e.: $F(\underline{X})$ is BBS iff there exist a set of two hyperplanes in $\mathbb{R}^n$ such that they separate $(F_{02} \cup F_-)$ from $(F_{01} \cup F_+)$ and $(F_{01} \cup F_-)$ from $(F_{02} \cup F_+)$ respectively.

When $F_{01} = \emptyset$, BBS turns into SBS.

Theorem 3: (3)

$F(\underline{X})$ is BBS iff there exist two Basis-functions $F_1(\underline{X})$ and $F_2(\underline{X}) \varepsilon (-1,1)$, such that one half their sum equals $F(\underline{X})$. The Basis-functions are two-valued threshold functions.

Proof: follows directly from def.3.

As may be seen from def.3, the set of SBS is a subset of the BBS functions. However, BBS functions are more difficult to analyse, as a good algorithm to obtain the partition of F_0 has not yet been found; and this partitions is needed to define the Basis-functions which realice the BBS function. If the partition is obtained, then realization suggestions from theorems 1 and 2 are also valid for the BBS function.

3. Use of Tables for the synthesis of SBS functions.

From theorem 1, to test SBS realizability of a function, is enough to test threshold realizability of its Basis-functions. Since Basis-functions are two-valued in (-1,1), characterizing and realization parameters of threshold functions in (-1,1) have been extracted from the general tables provided by the authors in a former contribution (4), and listed in Table I. This reduced Table, have only 5 and 41 entries for 2 and 3-place functions respectively, which makes the synthesis procedure very simple.

i) From $F(\underline{X})$ obtain $F_1(\underline{X})$ and $F_2(\underline{X})$

ii) Test, with aid of Table I, whether they are threshold functions or not.

iii) If at least one of them is not threshold, $F(\underline{X})$ is not SBS. If both are threshold, Table I provide the elements to obtain $\underline{w}_1$, $\underline{w}_2$, T_1, T_2.

iv) Function realization as shown in figs. 1 or 2.

4. Examples.

a) With use of Table I, test SBS realizability of the function described in the following truth - table:

X_1	X_2	X_3	F	F_1	F_2	X_1	X_2	X_3	F	F_1	F_2	X_1	X_2	X_3	F	F_1	F_2
-1	-1	-1	1	1	1	o	-1	-1	o	-1	1	1	-1	-1	o	-1	1
-1	-1	o	1	1	1	o	-1	o	1	1	1	1	-1	o	o	-1	1
-1	-1	1	1	1	1	o	-1	1	1	1	1	1	-1	1	1	1	1
-1	o	-1	-1	-1	-1	o	0	-1	o	-1	1	1	o	-1	o	-1	1
-1	o	o	-1	-1	-1	o	o	o	o	-1	1	1	o	o	o	-1	1
-1	o	1	1	1	1	o	o	1	o	-1	1	1	o	1	o	-1	1
-1	1	-1	-1	-1	-1	o	1	-1	-1	-1	-1	1	1	-1	-1	-1	-1
-1	1	o	-1	-1	-1	o	1	o	-1	-1	-1	1	1	o	o	-1	1
-1	1	1	-1	-1	-1	o	1	1	o	-1	1	1	1	1	o	-1	1

Truth - table, example a)

As $F_1(\underline{X})$ and $F_2(\underline{X})$ have been listed in the truth-table, the corresponding characterizing parameters may be calculated to test threshold realizability.

$F_1(\underline{X})$: $c_1=-6$, $c_3=6$, $p=7$, $m=20$

=> $F_1(\underline{X})$ ac → (12,6,6,7,20)

$F_2(\underline{X})$: $c_1=8$, $c_2=-12$, $c_3=6$, $p=19$, $m=8$

=> $F_2(\underline{X})$ ac → (12,8,6,8,19)

Since both ac-parameters are listed in Table I, $F(\underline{X})$ is SBS.

Following (4):

$F_1(\underline{X})$: (-1,-2,1, 3/2)

$F_2(\underline{X})$: (3,-4,2, -5/2)

Hence:

$\underline{w}_1 = (-1,-2,1)$; $T_1 = 3/2$

$\underline{w}_2 = (3,-4,2)$; $T_2 = -5/2$

Should a two gates realization be desired, eq.9 could be used, as $T_1>0$.

It may be shown that the minimum value of k is -3, H=6, and $-3 < L < 1.5$

Table I: Characterizing and realization parameters of 2 and 3-place ternary threshold functions in (-1,1)

c_1	c_2	c_3	p	m	w_1	w_2	w_3	T
6	2	-	4	5	2	1	-	o.5
6	0	-	3	6	1	0	-	o.5
4	4	-	3	6	1	1	-	o.5
4	2	-	2	7	2	1	-	1.5
2	2	-	1	8	1	1	-	1.5
18	6	2	13	14	4	2	1	o.5
18	6	0	12	15	2	1	0	o.5
18	4	4	12	15	3	1	1	o.5
18	4	2	11	16	5	2	1	1.5
18	2	2	1o	17	4	1	1	1.5
18	0	0	9	18	1	0	0	0.5
16	8	4	12	15	3	2	1	0.5
16	8	2	11	16	5	3	1	1.5
16	6	6	11	16	2	1	1	0.5
16	6	4	1o	17	4	2	1	1.5
16	4	4	9	18	3	1	1	1.5
16	2	2	8	19	4	1	1	2.5
14	10	6	13	14	4	3	2	0.5
14	10	4	11	16	5	4	2	1.5
14	10	2	10	17	4	3	1	1.5
14	8	8	12	15	3	2	2	0.5
14	8	6	10	17	4	3	2	1.5
14	8	4	9	18	3	2	1	1.5
14	6	4	8	19	4	2	1	2.5
14	4	2	7	20	5	2	1	3.5
12	12	6	12	15	2	2	1	0.5
12	12	0	9	18	1	1	0	o.5
12	10	8	11	16	5	4	3	1.5
12	10	2	8	19	4	3	1	2.5
12	8	8	9	18	3	2	2	1.5
12	8	6	8	19	4	3	2	2.5
12	8	2	7	20	5	3	1	3.5
12	6	6	7	20	2	1	1	1.5
12	6	0	6	21	2	1	0	1.5
12	4	4	6	21	3	1	1	2.5
10	10	10	10	17	1	1	1	0.5
10	10	4	7	20	2	2	1	1.5
10	8	4	6	21	3	2	1	2.5
10	6	2	5	22	4	2	1	3.5
8	8	4	5	22	2	2	1	2.5
8	6	2	4	23	3	2	1	3.5
6	6	6	4	23	1	1	1	1.5
6	6	0	3	24	1	1	0	1.5
6	4	4	3	24	2	1	1	2.5
4	4	2	2	25	2	2	1	3.5
2	2	2	1	26	1	1	1	2.5

The concept of Bilineal Separability is the "dual" from that of Polinomial Separability (5), where parallelismus is preserved, but separating hypersurfaces, not necessarily planes are used.

6. Bibliography:

(1) Moraga C.: "Funciones Ternarias de Umbral para dos Variables", Report EOSD 7105, Universidad Santa Maria, Valparaiso, Chile, (1971).

(2) Aibara T. and Akagi M.: "Enumeration of Ternary Threshold functions of three variables", IEEE Tr. EC-21, 402-407, (1972).

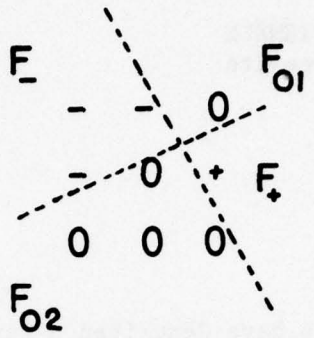
(3) Nazarala J.: "Estudio sobre Lógica de Umbral Ternaria", Thesis M.SC. in E.E., Universidad de Chile (Dec. 1973)

(4) Nazarala J. and Moraga C.: "Minimal Realization of Ternary Threshold Functions", 1974 Int. Symp. on Multiple-valued Logic. W.Va. USA

(5) Moraga C.: "Non-linear Ternary Threshold Logic" 1972 Int.Symp. on Multiple-valued Logic. SUNY at Buffalo, N.Y., USA

b) Test BBS realizability of the function whose Karnaugh map is shown below:

X_1 \ X_2	-1	0	1
-1	-1	-1	0
0	-1	0	1
1	0	0	0



It is easy to see that $F(\underline{X})$ is not SBS, as F_+ cannot be separated by an hyperplane. But letting F_{01} be vertex (-1,1), the function is BBS, as may be seen in the above graph.

It may be shown that:

$\underline{w}_1 = (-1, 2)$, $T_1 = 3/2$, $\underline{w}_2 = (2, 1)$, $T_2 = -1/2$

5. Conclusions:

The set of SBS functions is larger than that of threshold functions, and these functions have very simple realization with cascaded threshold logic.

Testing SBS functions may be done with aid of a very short table. This allows a conjecture that a table for 4-place two-valued threshold functions should still be of convenient size to explore 4-place SBS functions.

The set of BBS functions is even larger, but an algorithm for partition of F_0 has not yet been found.

IMPLEMENTATION OF A COMPLETE

TERNARY ALGEBRA WITH ELEMENTARY OPERATORS

APPLICATION TO TERNARY FLIP FLOP

par Daniel ETIEMBLE
PARIS VI Université
France

Michel ISRAEL
C.N.A.M.
France

I- INTRODUCTION

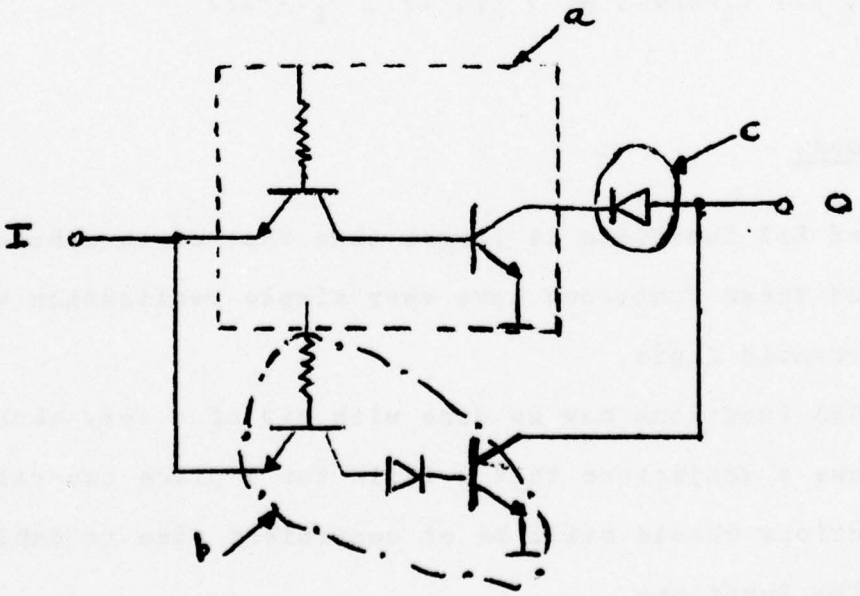
Authors have described a ternary inverter using TTL logical circuits (Series SN 7400) and diodes in a paper : "A new concept for ternary logic elements".[1]

In this paper, we show that the operators used allows to implement a complete ternary algebra.

We give a method for synthetizing combinatory functions $f(x_1.....x_n)$. We give the synthetizing of some sequential functions and the design of some ternary flip-flops (RS flip-flop, latch flip-flop, D flip-flop).

II- ELEMENTARY CIRCUITS

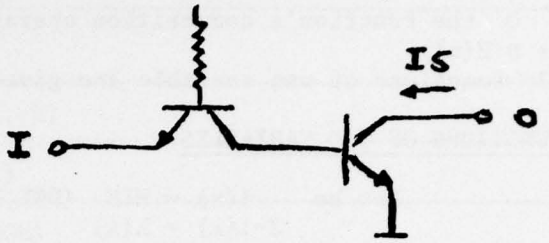
1 - Inverter



This inverter is built with three elements.
a) LTTL element (Low level TTL) :

The Threshold level is VT1 ie one base - emitter junction

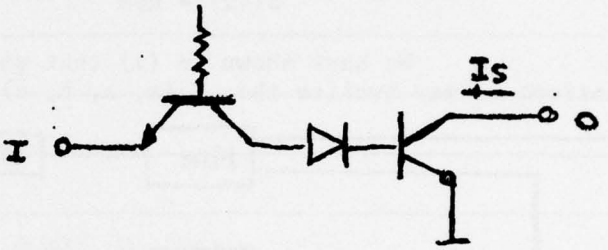
VI	IS	VS
VI < VT1	= 0	Haut
VT1< VI < VT2	≠ 0	Low
VI > VT2	≠ 0	Low



b) HTTL element (High Level TTL) :

The Threshold level is VT2 ie two base emitter junctions

VI	IO	VO
VI < VT1	= 0	High
VT1< VI < VT2	= 0	High
VI > VT2	≠ 0	Low



c) Diode D :

When II ≠ 0 then VO = VI + one diode junction.

VI	II = IO	VO
Low	≠ 0	Medium
Medium	≠ 0	High
High	≠ 0	High
VI	= 0	High



Each one of these elementary operators realize a defined ternary function, used separatly, or combined.

- LTTL operator.

x	-	0	+
L (x)	+	-	-

- HTTL operator

x	-	0	+
H (x)	+	+	-

- Diode operator

x	-	0	+
D (x)	0	+	+

2 - MIN Operator.

We get the min operator by wiring together the outputs of two L(x), H(x), or D(x) operators ("wired or" technology).
- The set of L(x), H(x), and D(x) operators is functional.

III - FUNCTIONS OF ONE VARIABLE

There are $3^3 = 27$ functions of one variable :

f a-1, a_0, a_1

Let be 0 the function's composition operator.

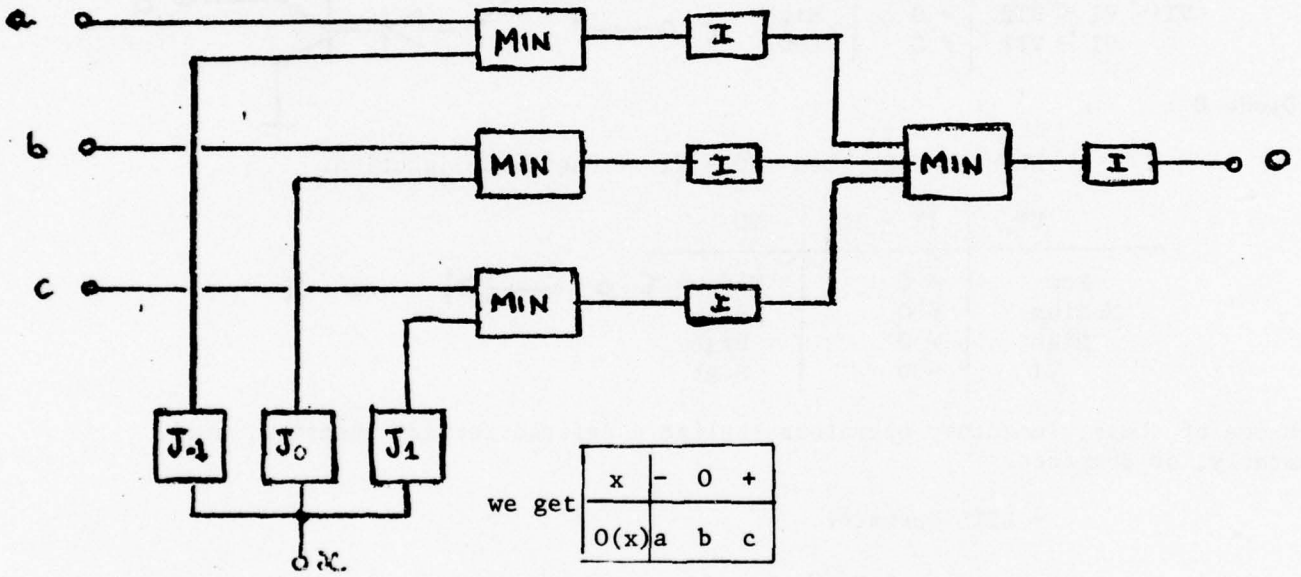
D 0 H = D(H(x))

The 27 functions of one variable are given in the following array :

IV - FUNCTIONS OF TWO VARIABLES

Let be
$$I(x) = MIN\ (D\emptyset L, H) \quad (f8)$$
$$J\text{-}1(x) = L(x) \quad (f5)$$
$$JO(x) = MIN\ (L\emptyset L, H) \quad (f\text{-}7)$$
$$J1(x) = H\emptyset H \quad (f\text{-}11)$$

We have shown in (1) that with the MIN, J - 1(x), Jo(x), $J_1(x)$ functions we may realize the T (x, a, b, c) function of Lee and Chen (2) !



we get

x	-	0	+
0(x)	a	b	c

The ternary element :

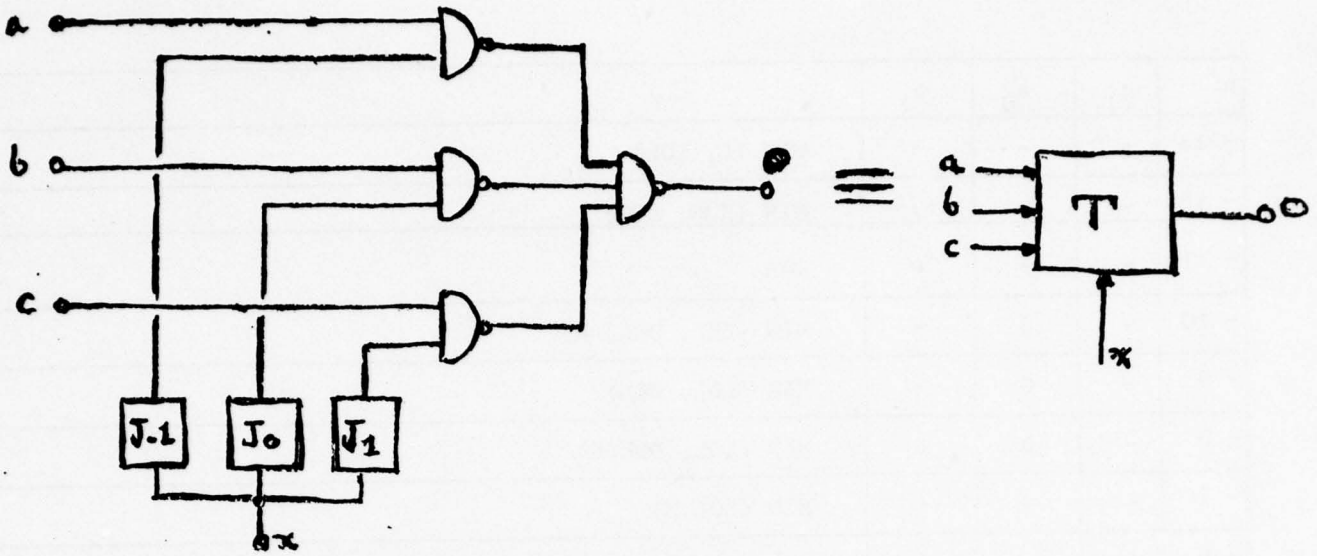


is equivalent to :



Then we get the following design of the T operator.

n°	a_1	a_0	a_1	
- 13	-	-	-	MIN (L, LOL)
- 12	-	-	0	MIN (HOH, DOL)
- 11	-	-	+	HOH
- 10	-	0	-	MIN (LOL, DOL, H)
- 9	-	0	-	MIN (LOL, DOL)
- 8	-	0	+	MIN (LOL, DOHOH)
- 7	-	+	-	MIN (LOL,H)
- 6	-	+	0	MIN (LOL, DOH)
- 5	-	+	+	LOL
- 4	0	-	-	MIN (D,L)
- 3	0	-	0	MIN (DOL,H) 0 MIN (0, DOH)
- 2	0	-	+	MIN (DOL,H) 0 MIN (D,H)
- 1	0	0	-	MIN (DOHOH, H)
0	0	0	0	MIN (D, DOL)
1	0	0	+	DOHOH
2	0	+	-	MIN (D,H)
3	0	+	0	MIN (D,DOH)
4	0	+	+	D
5	+	-	-	L
6	+	-	0	MIN (DOL,H) 0 MIN (LOL, DOH)
7	+	-	+	MIN (DOL,H) 0 MIN (LOL, H)
8	+	0	-	MIN (DOL, H)
9	+	0	0	DOL
10	+	0	+	MIN (DOL,H) 0 MIN (LOL, DOL, H)
11	+	+	-	H
12	+	+	0	DOH
13	+	+	+	DOD



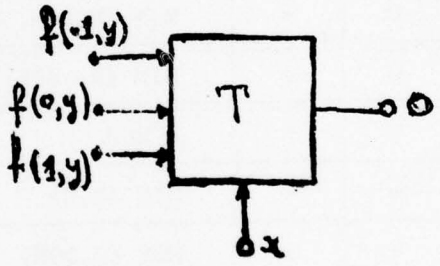
To implement the two variable functions, we use the Lee and Chen algorithm of decomposition (2)

$$f(x_n + 1, x_n, \ldots\ldots x_1) = T\,(x_n + 1,\ f\,(-1, x_n \ldots\ldots x_1))$$
$$f\,(0, x_n \ldots\ldots x_1))$$
$$f\,(+1, x_n \ldots\ldots x_1))$$

The synthesis of any two variables function is made by :

x \ y	-	0	+
-	α	α	α
0	α	α	α
+	α	α	α

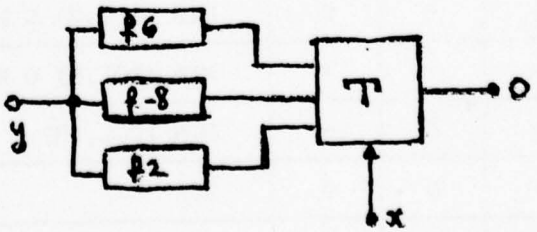
f(-1,y) f(1,y)
f(0,y)



Example

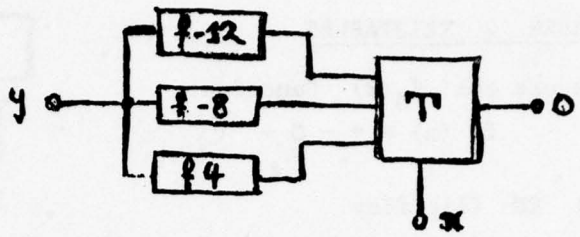
a) Cyclic Addition

x \ y	-	0	+
-	+	-	0
0	-	0	+
+	0	+	-
	f_6	f-8	F_2



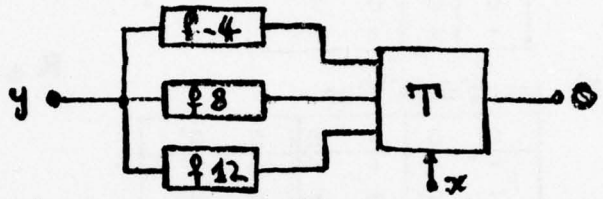
b) Limited addition

y \ x	-	0	+
-	0	-	0
0	-	0	+
+	0	+	+
	f-12	f-8	f4



c) Comparator

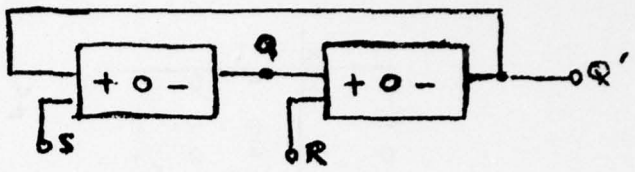
y \ x	-	0	+
-	0	+	+
0	-	0	+
+	-	-	0
	f-4	f8	f12



NB : According to technological considerations, we may simplify the design of these circuits i.e the number of necessary elements.

APPLICATION TO TERNARY FLIP FLOPS

I - ORDER 2 TRISTABLES

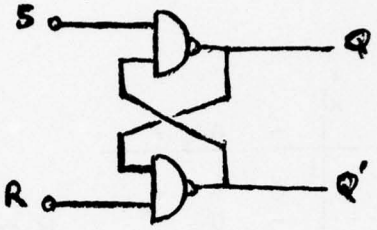


We use the $f_8(x)$ function.

$$f_8(x) = + - 0 -$$

1) RS flip flop

S	R	Q	Q'
+	+	Memory	
+	-	-	+
0	0	0	0
-	+	+	-



2) Latch flip flop

C	D	S	R	Q	Q'
-	-	+	+	Q	Q'
-	0	+	+	Q	Q'
-	+	+	+	Q	Q'
+	-	+	-	-	+
+	0	0	0	0	0
+	+	-	+	+	-

We must implement the following synthesis :

C \ D	-	0	+	
-	+	+	+	f13
0	+	+	+	
+	+	0	-	f+8

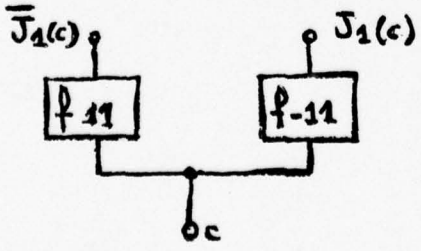
Set Input

C \ D	-	0	+	
-	+	+	+	f13
0	+	+	+	
+	-	0	+	f-8

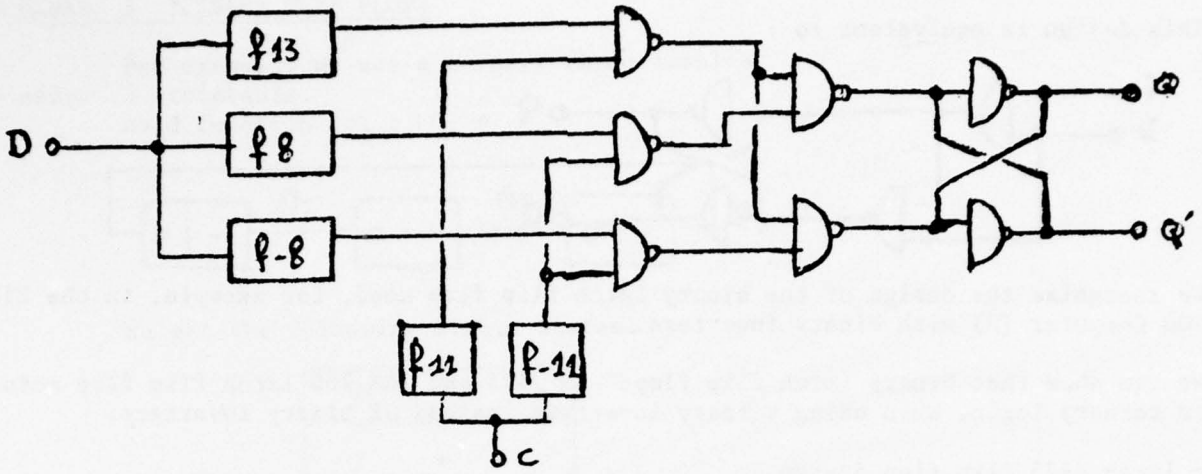
Reset Input

NB : The clock C takes only the - and + states.

We may simplify this circuit using $\overline{J1}$ instead of Jo and J-1 :



So we get the following circuit



Simplifications :

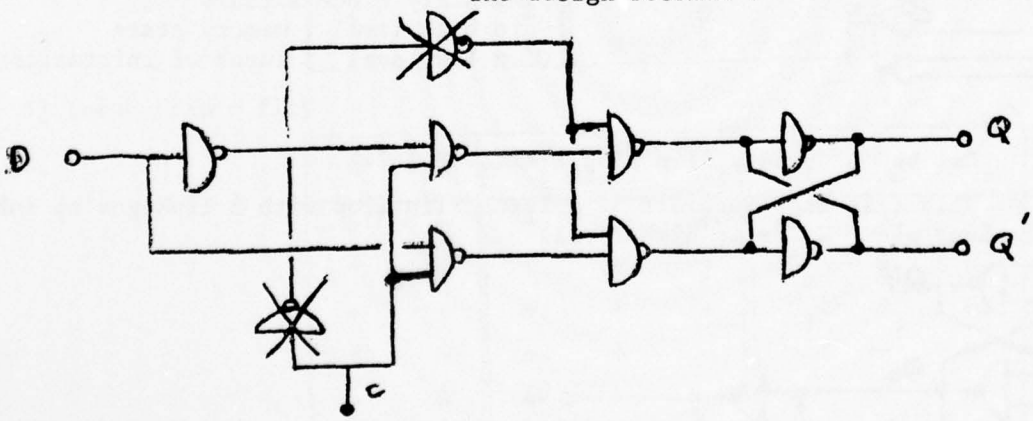
f13 : This function is only a high level source. With the used technology, it will be a floating input emitter transistor.

f8 : ternary inverter.

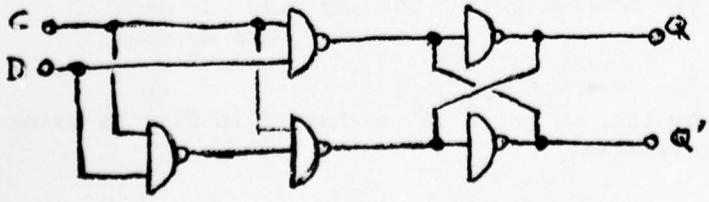
f-8 : non inverter. This element can be omited when D information is a quantified signal.

As C only takes two levels, the f11 function can be implemented.
f8.

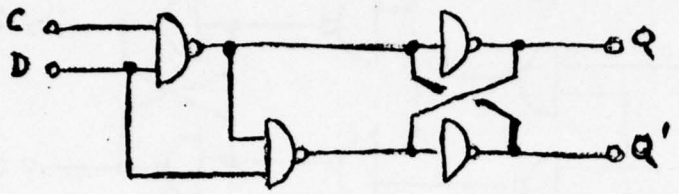
The design becomes :



So :



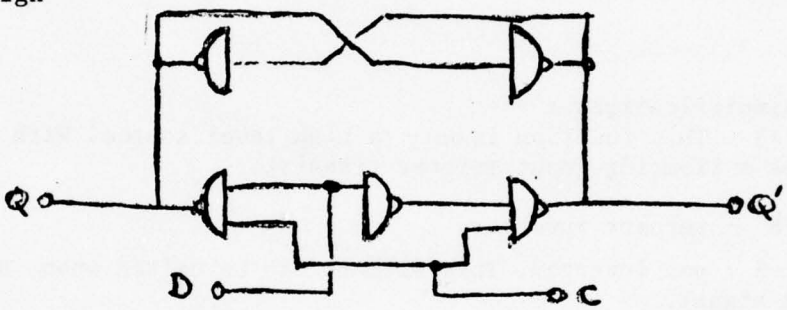
This design is equivalent to :



We recognize the design of the binary latch flip flop used, for example, in the Elbit 100 Computer (3) with binary inverters.

We can show that binary latch flip flops SN 7475 and IBM 360 latch flip flop actually in ternary logic, when using ternary inverters instead of binary inverters.

- latch 7475 flip flop design



- latch IBM 360 flip flop design.



This latch flip flop works with complementary clock signals
C in high level : memory state
C in low level : input of information

3) D flip flop

Let be D binary flip flop (Texas SN 7474).

This flip flop consists of a latch flip flop with 3 linkages to inhibit D variations when C is in high level.

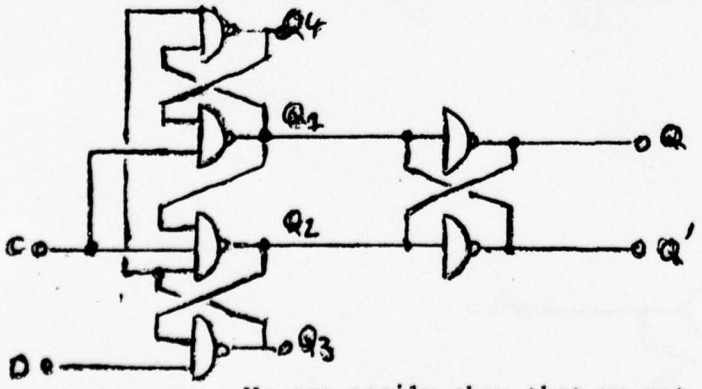
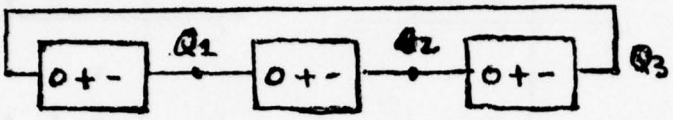


figure A

We can easily show that we get a D ternary flip flop by using ternary inverters instead of binary ones.

II - ORDER 3 TERNARY FLIP FLOPS

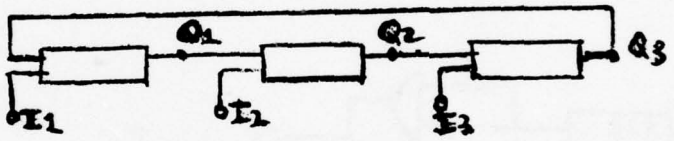
For example, we use a ternary three stables
- order 3 tristable.
used function f_2 : 0 + -



We get the following stable states.

O_1	O_2	O_3
0	+	-
+	-	0
-	0	+

Ternary flip flop



I_1	I_2	I_3	O_1	O_2	O_3
+	+	+	Memory		
+	+	-	+	-	0
+	-	+	-	0	+
-	+	+	0	+	-

1) Latch flip - flop

C	D	I_1	I_2	I_3	O_1	O_2	O_3
-	-	+	+	+	O_1	O_2	O_3
-	0	+	+	+	O_1	O_2	O_3
-	+	+	+	+	O_1	O_3	O_3
+	-	+	+	-	+	-	0
+	0	+	-	+	-	0	+
+	+	-	+	+	0	+	-

When C is low, we get a memory state.
When C is High, the output follows the data input as long as the clock remains high

$$O_1 = f_6(O_2), \quad O_2 = D, \quad O_3 = f_2(O_2)$$

Ternary latch flip flop

C \ D	-	0	+
-	+	+	+
0	+	+	+ f13
+	+	+	-f11

Input 1

C \ D	-	0	+
-	+	+	+
0	+	+	+ f13
+	+	-	+f7

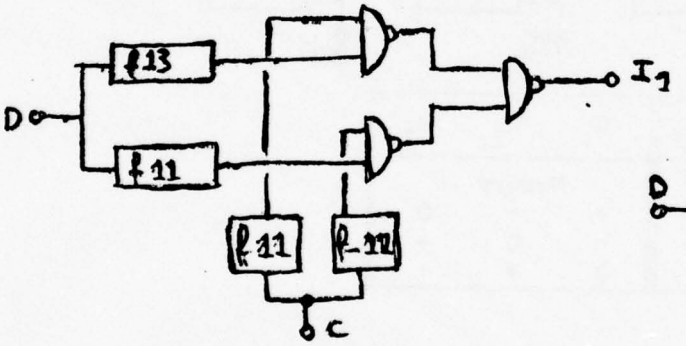
Input 2

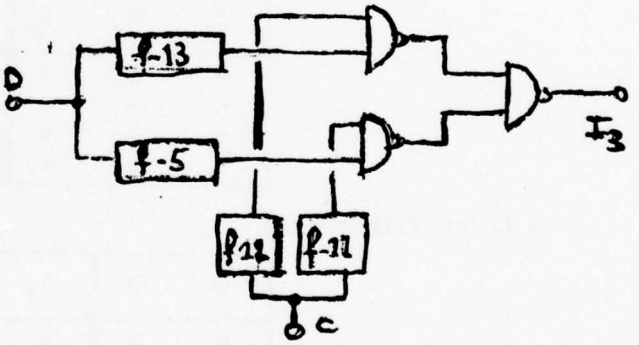
C \ D	-	0	+
-	+	+	+
0	+	+	+ f13
+	-	+	+f-5

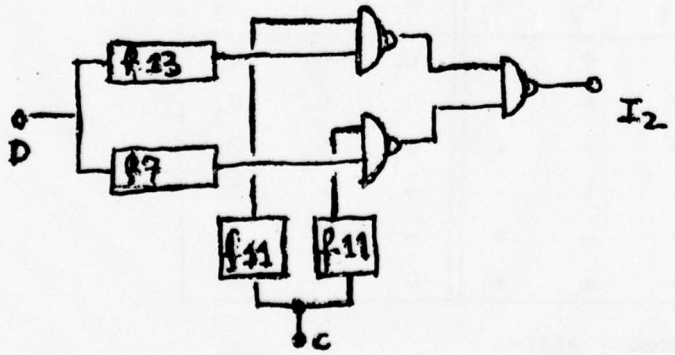
Input 3

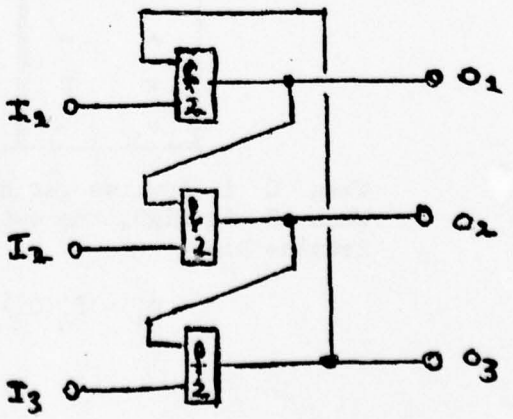
As C = 0 and C = - states give yhe same E_1, E_2, E_3 inputs we can simplify the circuits in the same way as order 2 latch.

Let be the circuits to get I_1, I_2, I_3 :

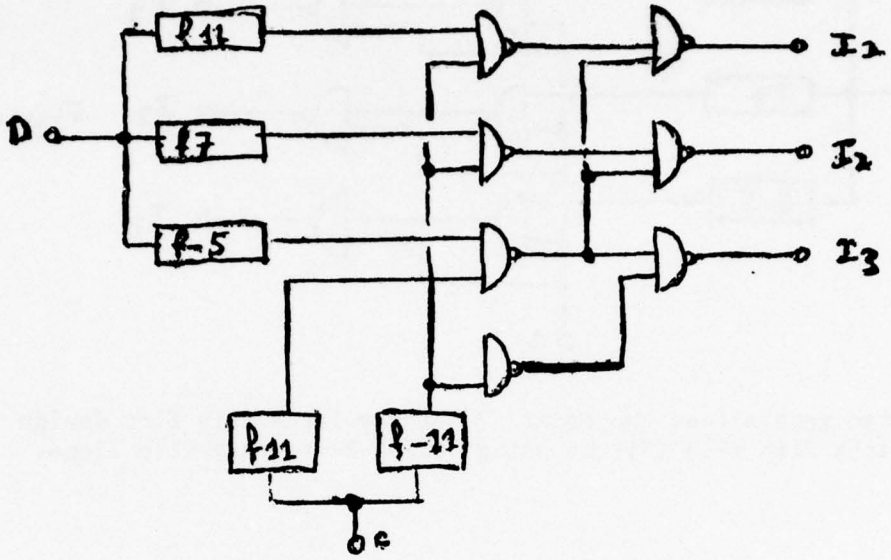




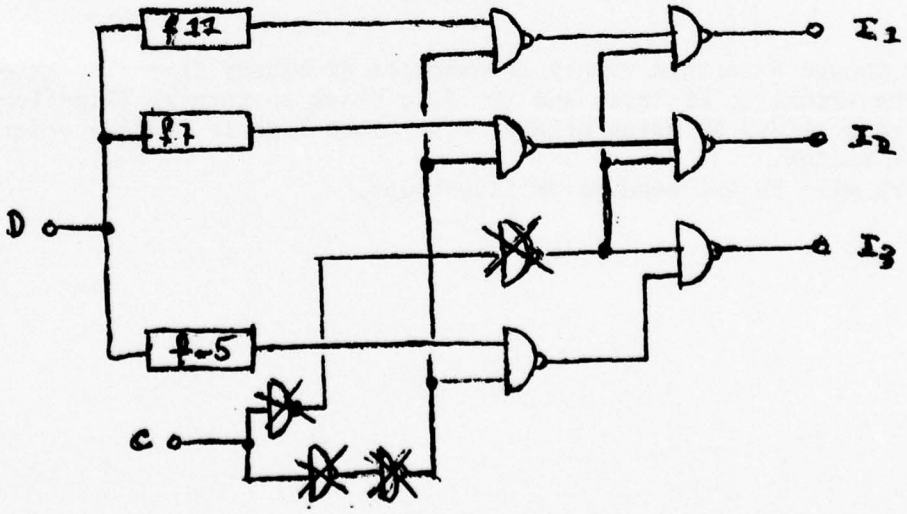




The design simplification gives :



After simplification, we get the following design.
(The clock having only two states, we use f_{18} instead of f11)



So, we get :

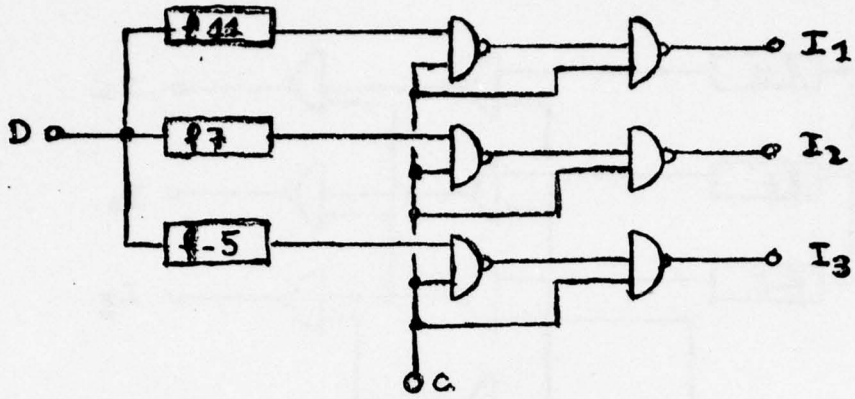
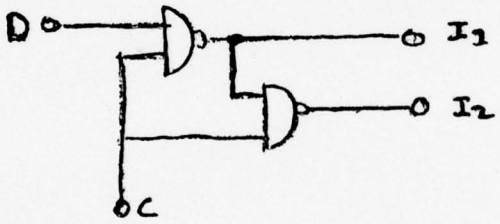


Figure 8

NB. This design generalizes the order 2 binary latch flip flop design (Elbit latch flip flop (3)) by using order 3 ternary flip flops.



CONCLUSION

We have showed a certain number of examples of binary flip-flop extensions, particularly the extension of latch and D flip flops to ternary flip-flops. These examples can be generalized by using others f(x) functions to realize order 2 or order 3 three states.
This work will be extended to JK flip-flops.

[1] " A New Concept for ternary logic elements"
1974 Symposium on multiple valued logic.
Daniel ETIEMBLE, Michel ISRAEL

[2] " Several-Valued Combinational Switching Circuits"
C.Y. LEE, W.H. CHEN AIEE JULY 1956 pp. 278-283.

[3] "Contribution à l'étude et à la modélisation des circuits logiques"
Daniel ETIEMBLE. Thèse docteur ingénieur PARIS VI.1974

SUMMARY

SOME MULTI-VALUED APPROACHES TO TWO-VALUED SWITCHING PROBLEMS†

G. Metze
University of Illinois, Urbana
USA

Multi-valued logics have obvious applications in situations where it is convenient or necessary to let a variable assume more than two states, as, for example, in non-binary arithmetic units. Some new design methods for combinational and sequential networks which are more or less straightforward extensions of binary design methods have been proposed. These will be reviewed briefly, with emphasis on the issues which cause them to differ from binary or binary-coded multi-valued methods.

The main part of the talk is concerned with applications of multi-valued logics to problems which can be treated more conveniently by direct use of multi-valued methods rather than by binary or binary-coded methods. Specific examples include the detection of hazards in binary networks, the design of hazard-free networks and the diagnosis of faults in binary networks.

---

† This paper will appear in the book Modern Uses of Multiple-Valued Logic.

ON THE EFFICIENCY OF TERNARY ALGORITHMS FOR MULTIPLICATION AND DIVISION

Amnon Barak
Hebrew University of Jerusalem
Israel

Ellen Aron
Hebrew University of Jerusalem
Israel

ABSTRACT

This paper deals with algorithms for multiplication and division using ternary arithmetic. Several algorithms are developed and compared to similar algorithms using binary arithmetic. The method used is a normalization process of an operand to a specific constant. All of the algorithms converge and the rates of convergence are used as a measure of efficiency.

I. INTRODUCTION

When computers were first being designed and built, it was due primarily to hardware considerations that binary arithmetic and binary logic were adapted for use. Although the industry has gone through several generations of hardware (there exist n-flop devices which have n stable states) and there exists a well-based theory of multi-valued logic, binary arithmetic and binary logic are dominant.

It has been shown that the radix for the most efficient representation of numbers on a computer is e, the base for natural logarithms [1]. The closest integral value to e is 3.

In this paper, we develop several algorithms for multiplication and division using ternary arithmetic based on similar algorithms which were developed by De Lugish [2] using binary arithmetic. We use a measure of efficiency called the shift average which was developed by Robertson [1] to compare these algorithms. It is particularly useful to study the algorithms for multiplication and division because the method of normalization used in these algorithms can also be used with very slight modification for computing the natural logarithm function and other basic arithmetic and trigonometric functions which are currently being done by software.

In section II, we define the measure of efficiency which we use. In section III, we introduce the method of normalization upon which all of the division algorithms are based. We present several algorithms for division using binary arithmetic and discuss their efficiency. In section IV, we present the balanced ternary representation for numbers and develop several algorithms for division using ternary arithmetic. The normalization process for multiplication is given in section V and the binary multiplication algorithm is discussed. Section VI contains a comparison of several ternary multiplication algorithms.

II. THE MEASURE OF EFFICIENCY IN BINARY: THE SHIFT AVERAGE

The property which characterizes the class of algorithms which we are considering here is that they sequentially check each bit in the argument and, only if it is on, perform some operation.

Checking whether or not a bit is on is, relatively, not time-consuming and

the number of these checks is equal to the number of bits in the argument. However, performing an operation is time-consuming and one algorithm which performs fewer operations than a second algorithm to reach the desired result is clearly more efficient.

A measure of efficiency for these algorithms, called the shift average, has been defined by Robertson [1]. If we define P to be the probability of a zero bit, then the shift average, <s> is defined as follows:

$$<s> = \frac{1}{(1 - P)} \quad .$$

That is, if the current bit is on, then only after <s> bits, on the average, to the right (right because we are dealing with normalized numbers) will another bit be on. Therefore, to increase the shift average of an algorithm, the argument must be represented in a way which minimizes the number of non-zero bits.

Using the ordinary binary representation for numbers, the possible values which a bit can assume are 0 and 1. Thus, the probability of a zero digit is 50% ($P = \frac{1}{2}$) and <s> = 2. This is the non-redundant (or conventional) representation as the number of possible values for each digit is equal to the radix - in this case, 2. A given number can be represented in one, unique way.

When a binary number is represented using a redundant representation, each digit may assume one of three different values. One possible set of values is: 1,0,-1. If we assume equal probability for each of the possible values for a digit, then the probability of a zero digit is only 33%. There is more than one way to represent any given number. However, the various representations differ in some of their properties, in particular, the probability of zero is not necessarily 33% and therefore the shift average is not necessarily $\frac{3}{2}$. For example, a number represented using one of these redundant representations (called the 'canonical' recoding) has at least one zero digit between every two non-zero digits and the shift average for this representation is 3. Other redundant representations result in shift averages of 1 or 2 [1].

III. EFFICIENCY OF BINARY DIVISION ALGORITHMS

We are interested in finding the value $\frac{Y}{X}$, where both X and Y are normalized binary floating point numbers. If we multiply both X and Y by some number, D, such that $X \cdot D = 1$, then $Y \cdot D = \frac{Y}{X}$. Clearly, $D = \frac{1}{X}$. In these algorithms, the number D is actually a product of n numbers, d_i, where $1 \leq i \leq n$, and n is one less than the number of bits per number.

The original argument, X, lies in a certain range which is symmetric around 1, $(1 - z_1, 1 + z_2)$. We multiply X by some number, d_i, such that $X d_1$ is within a second range $(1-z_2, 1 + z_2)$ which is smaller than the first range $(1 - z_1, 1 + z_1)$ and still symmetric around 1. That is

$$(1 - z_2) < (1 - z_1) \cdot d_1 < (1 + z_2)$$

and

$$(1 - z_2) < (1 + z_1) \cdot d_1 < (1 + z_2).$$

This process of multiplication continues until X has been multiplied by all of the d_i such that, after the n-th step:

$$(1 - z_{n+1}) < X \cdot d_1 \cdot d_2 \cdot \ldots \cdot d_n < (1 + z_{n+1}).$$

Let $x_1 = X$ then $x_{i+1} = x_i \cdot d_i$. x_{i+1} is closer to 1 than is x_i. We say that X, the divisor, is 'normalized' to 1. Y is also multiplied by all of the $d_1 (y_{i+1} = y_1 \cdot d_1)$ such that, at the end, $\frac{Y}{X} = y_{n+1}$.

We must now choose the d_i and the z_i. If we choose the z_i to be 2^{-i} then our final result, x_{n+1}, is in the range $(1-2^{-n+1}) < X \cdot d_1 \cdot d_2 \cdot \ldots \cdot d_n < (1+2^{-n+1})$, which is a '1' followed by at least n zeroes.

Two types of binary algorithms have been developed which use this technique: non-redundant and redundant. Using the non-redundant recoding there are two possible values for each digit since the radix is 2.

It is convenient to define the d_i in terms of another constant s_i:

$$d_i = 1 + s_i \cdot 2^{-i}$$

where

$$s_i = \begin{cases} -1 & \text{for } x_i > 1 \\ +1 & \text{for } x_i < 1. \end{cases}$$

Here every multiplication is really just a'shift'followed by an 'addition' operation or a 'subtraction' operation. The series s_i is a non-restoring recoding of $\frac{1}{X}$. Since $s_i = \{1, -1\}$, there are no zero digits and the shift average is 1. The quotient is conventionally coded.

If we define s_i to be

$$s_i = \begin{cases} 0 & \text{for } x_i > 1 - 2^{-i} \\ 1 & \text{for } x_i \leq 1 - 2^{-i} \end{cases}$$

then the shift average is increased to 2 as $s_i = \{0,1\}$ and $P = \frac{1}{2}$. The quotient is also conventionally coded.

By allowing three digital values for s_i even though the radix is 2, we can introduce redundancy into the representation.

We may define s_i to be

$$s_i = \begin{cases} 1 \text{ for } x_i < 1 - c \cdot 2^{-i} \\ 0 \text{ otherwise} \\ -1 \text{ for } x_i > 1 + c \cdot 2^{-i} \end{cases} \text{ for some constant, } c.$$

Intuitively, one might expect that because $s_i = \{1,0,-1\}$ then P must be equal to $\frac{1}{3}$. But it is not.

At each step the range is essentially divided into three unequal intervals - we divide the range into four equal parts and define:

the lowest part is the first interval,
the next two parts for the second interval, and
the remaining part is the third interval.

That is, the second interval is twice as large as each of the other two intervals. The intervals correspond to the s_1: the first interval corresponds to $s_i = 1$, the second corresponds to $s_i = 0$, and the third interval corresponds to $s_i = -1$. Since the second interval is exactly half of the range, one might expect that P would be equal to $\frac{1}{2}$. But it is not.

The value of P is very dependent on the value of c. If we define $c = \frac{3}{4}$, then $P = \frac{2}{3}$, and the shift average, $s = 3$! When $c = \frac{3}{4}$, every non-zero quotient digit is followed by at least 1 zero digit, i.e., the quotient is in canonical form. This occurs because when the quotient digit is non-zero, then, in the partially normalized divisor where the current digit has just become zero, the next digit is now also equal to zero.

The numbers X and Y are normalized binary floating point numbers which means that thev are in the range $[\frac{1}{2}, 1)$. The range is clearly not symmetric around 1. However the algorithm is meant only for numbers in a range which is symmetric around 1. So we must define an initialization step.

In order for the 'multiplication' $x_{i+1} = x_i \cdot d_i$ to be a 'shift' followed by an 'addition' or a 'subtraction' operation, the number x_i must be represented with one digit to the Ieft of the binary point which must be a '1'. The divisor is represented in canonical recoding so the next digit must be a zero. So the range for the initial divisor, X, is really $(\frac{2}{3}, \frac{4}{3})$ as:

$$1.0\overline{1}0\overline{1}\overline{1}\cdots = \frac{2}{3} \quad \text{and} \quad 1.0101\cdots = \frac{4}{3}.$$

We use the symbol 'T' to represent '-1' or a '$\overline{1}$' for convenience of representation (see discussion in Frieder and Luk [3]).

In order to get the normalization divisor into this first range, we could define $x_1 = x_0 \cdot d_0$, where $x_0 = X$ and

$$d_0 = \begin{cases} 2 & \text{for } X < \frac{2}{3} \\ 1 & \text{for } X \geq \frac{2}{3} \end{cases}$$

But this involves a full-register comparison. Instead of this, we define

$$d_0 = \begin{cases} 2 & \text{for } X < \frac{3}{4} \\ 1 & \text{for } X \geq \frac{3}{4} \end{cases}$$

and the range of x_1 is $(\frac{3}{4},\frac{3}{2})$. Although this is not symmetric around 1, it is usable.

A number consisting of 43 bits is equivalent to 13 decimal digits of accuracy. So we need 42 iterations in order to get the canonical recoded normalized divisor 'normalized' to 1. But since $\langle s \rangle = 3$, we expect that only 14 operations will actually be performed. Statistically, we have shown that, indeed, on the average only 14 operations are performed.

IV. EFFICIENCY OF TERNARY DIVISION ALGORITHMS

We shall use a balanced representation for ternary numbers. Any number can be uniquely represented as a balanced ternary number:

$$a_n a_{n-1} \cdots a_1 a_0 \cdot a_{-1} a_{-2} \cdots a_{-k} = \sum_{i=-k}^{n} a_i \cdot 3^i \text{ where } a_i \in \{-1,0,1\} .$$

Since we are using three digital values (1,0,-1) and the radix is three, our representation is non-redundant.

We start with a balanced ternary number, X, in the range $(\frac{1}{2},\frac{3}{2})$. That is, X has one digit to the left of the ternary point which is a '1' and the digits to the right of the ternary point (the fractional digits) are either 0,1, or -1. If there were an infinite number of fractional digits and they all were -1's, then the number represented would be $\frac{1}{2}$, or, if they all were 1's the number would be $\frac{3}{2}$.

We wish to normalize X to 1. This is done by multiplying X by n factors, d_k, $1 \leq k \leq n$, such that after k multiplications, trits 1 to k in the product are all equal to zero. Let trit k be the first non-zero fractional trit. If trit k is 1, then $d_k = 1 - 3^{-k}$, or, if it is -1, then $d_k = 1 + 3^{-k}$. (For the sake of consistency, we could say that if trit k is 0, then $d_k = 1$. This means that we shift x_{k-1} by k positions to the right and either add this shifted value to x_{k-1} or subtract it from x_{k-1}.

More formally, for $1 \leq k \leq n$,

$d_k = 1 + s_k \cdot 3^{-k}$ where $x_{k+1} = x_k \cdot d_k$ where $x_1 = X$, and $x_{n+1} = 1.0$.

$$s_k = \begin{cases} 1 & \text{for } x < 1 - \frac{1}{2}\,3^{-k} \text{------- trit } k = -1 \\ 0 & \text{otherwise ---------- trit } k = 0 \\ -1 & \text{for } x > 1 + \frac{1}{2}\,3^{-k} \text{------- trit } k = 1 \end{cases} .$$

Since s_k is the complement of the k-th trit we can define

$$d_k = 1.0 \cdots 0\ \overline{x_k^{(k)}}\ 0 \cdots 0$$

where $x_k^{(j)}$ is the j-th digit after the ternary point in x_k. The algorithm for $\frac{Y}{X}$, which was developed by Barak [4], is as follows:

ALGORITHM I: TERNARY DIVISION USING THREE INTERVALS FOR EACH RANGE

1. set $x_1 = X$, $k = 1$, $y_1 = 1$;
2. set $d_k = 1.0 \cdots 0\ \overline{x_k^{(k)}}\ 0 \cdots 0$; then $x_{k+1} = x_k \cdot d_k$ and $y_{k+1} = y_k \cdot d_k$;
3. if $x_{k+1} < 1 - (\frac{1}{2}) \cdot 3^{-k}$ or $x_{k+1} > 1 + (\frac{1}{2}) \cdot 3^{-k}$ (i.e. if the k-th trit after the ternary point is still non-zero) then go to step 2 else set $k = k + 1$; if $k = n$ then we are done and $\frac{Y}{X} = y_{k+1}$, otherwise go to 2.

At step $k + 1$, the first k fractional digits of x_k are zero, x_k is in the range $(1 - (\frac{1}{2}) \cdot 3^{-k},\ 1 + (\frac{1}{2}) \cdot 3^{-k})$, and when the k+1 -th trit is set to zero, the number x_{k+1} will be in the range

$$(1 - \tfrac{1}{2}\,3^{-(k+1)}\ ,\ 1 + \tfrac{1}{2}\,3^{-(k+1)}) .$$

In effect, we divide the range $(1 - \frac{1}{2}\,3^{-k}\ ,\ 1 + \frac{1}{2}\,3^{-k})$ into three intervals, corrseponding to the value of the k-th trit; then the mapping is the result of multiplying the numbers in each interval by the appropriate constant:

$$\text{trit } k = -1\text{:}\quad (1 - \tfrac{1}{2}\,3^{-k},\ 1 - \tfrac{1}{2}\,3^{-(k+1)})\ (1 + 3^{-k})$$

$$\text{trit } k = 0\text{:}\quad (1 - \tfrac{1}{2}\,3^{-(k+1)}, 1 + \tfrac{1}{2}\,3^{-(k+1)})\ (1) = (1 - \tfrac{1}{6}\,3^{-k}, 1 + \tfrac{1}{6}\,3^{-k})$$

$$\text{trit } k = 1\text{:}\quad (1 + \tfrac{1}{2}\,3^{-(k+1)},\ (1 + \tfrac{1}{2}\,3^{-k})\ ,\ (1 - 3^{-k})$$

However, after the mapping is done, some of the numbers originally in the outer intervals are not mapped successfully into the inner interval. That is, after step k, trit k is still not equal to zero. This occurs if a carry develops during the addition or subtraction and is propagated to the k-th position. If we repeat the same mapping once more (do step k again), then all of the numbers are mapped into the inner interval, and trit k is forced to zero.

$$(1 - \frac{1}{2}\,3^{-k} - \frac{3}{2}\,3^{-2k},\ 1 - \frac{1}{2}\,3^{-k})\ (1 + 3^{-k}).$$

The lower bound for the region of number which are not successfully mapped by one operation can be considered as an error factor of the order $0(3^{-2k})$. This error is significant only when k is small because for a large k the error appears in digits which are not significant to our accuracy. The remapping is necessary in order to correct the error. The details of the mapping and the proof of convergence are given in [4]. Thus, trit k is checked once in order to select the appropriate s_k, we do in the operation, and then trit k is checked again, and it is possible that the operation will be performed again.

A number consisting of 28 ternary digits is equivalent to 13 decimal digits of accuracy. One trit is on the left of the ternary point which leaves 27 trits to be checked and zeroed, i.e., 27 iterations of the algorithm must be performed. However, we must modify the iteration count in light of two additional considerations. Firstly, no operation is performed when a digit is zero (so that the iteration does not do anything). Hence, we can subtract the number of zero digits from the iteration count. Secondly, we must add 1 to the iteration count for every repetition because then an additional operation is performed in an effort to zero out a particular digit. Thus, the shift average for this algorithm can be computed by dividing 27 by the modified iteration count.

Since we are using a non-redundant representation for balanced ternary numbers, we assume that we must check a trit twice for 66% of the time (the probability of a non-zero trit is 66%). The expected shift average is therefore

$$\langle s \rangle = \frac{1}{1 - \frac{1}{3}} = \frac{3}{2}\ .$$

We have shown experimentally that this assumption is correct: $\frac{1}{3}$ of the trits are zeroes.

We have also shown that the number of repetitions (when step k must be performed twice) is .3. That is, statistically, there is 1 repetition for every 3 numbers normalized.

In the non-redundant case, there are three different values for s_k which corrsepond to the three possible values which each digit can assume. Each range is divided into three intervals.

There are several independent variables in the redundant case:

1) the number of intervals into which each range is divided
2) the boundary points between the different intervals, and
3) the values for s_k,

and each of these gives us one degree of freedom.

Although the number of possible combinations for these variables is very large with three degrees of freedom, we have tried to limit ourselves to using only simple constants. A 'simple constant' can be represented as a balanced ternary number using a maximum of digits, and therefore a maximum of two non-zero digits. Thus there are 3^2 possible 'simple constants' from which to choose. The reason for this limitation is two-fold.

1) it limits the number of digits to be checked when selecting s_k to 2, thereby avoiding, for example, a full-register comparison, and
2) it limits the number of addition/subtraction operations to be performed at any iteration to a maximum of 2.

However, when the simple constant forces us to perform two operations at an iteration in order to get digit k to become zero, we would like to be sure that digit k+1 will also become zero at the same time. That is, we want every operation to give us one zero digit, so that every iteration will give us at least one zero digit and possibly two.

To get the smallest number of intervals we can define using a redundant representation, the outer intervals of the original range are divided into two intervals each, giving five intervals for the entire range.

We wish to map each of the four outer intervals of the original range $(1 - \frac{1}{2}\,3^{-k},\ 1 + \frac{1}{2}\,3^{-k})$ to two levels down, i.e., to $(1 - \frac{1}{2}\,3^{-k-2},\ 1 + \frac{1}{2}\,3^{-k-2})$.

Let a be the point which divides the lower interval into two parts and let b be the point which divides the upper interval into two parts. There do not exist unique values for a and b with which the mapping function can achieve the desired result. Thus, it is impossible to get one zero digit for every operation when the original range is divided into only five intervals.

When each of the original intervals of the original range is divided into three intervals each, there are seven intervals for the entire range. Using seven intervals, it is definitely possible to map successfully two levels down as we show below in our proof that this algorithm converges.

Let R_i be the i-th interval of the range. The intervals are defined as follows (digits 0 through k-1 after the ternary point are zero already):

		corresponding simple constants
In R_1, digits k and k+1 after the ternary point are	TT	$1.1 = \frac{4}{3}$
R_2	T0	$1.0 = 1$
R_3	T1	$1.T = \frac{2}{3}$
R_4 digit k = 0		
R_5	1T	$T.1 = -\frac{2}{3}$
R_6	10	$T.0 = -1$
R_7	1T	$T.T = -\frac{4}{3}$

The simple constant by which the number in each interval are multiplied is simply the number which is the complement of digits k and k+1 with a ternary point between them.

Just as it was necessary to define an initialization step for the redundant binary algorithm, it is also necessary for the redundant ternary algorithms but for a different reason.

Again, in order for the multiplication to really be a 'shift' followed by an addition/subtraction operation, the operand must have a '1' to be the left of the ternary point and the rest of the digits to the right of the ternary point. Thus, the obvious initial range is (1.TTT···, 1.111···) which is $(\frac{1}{2}, \frac{3}{2})$. However, the numbers in a rather large interval in this initial range do not map successfully into the second range $(\frac{5}{6}, \frac{7}{6})$. The algorithm does not converge for these numbers. During the discussion of error factor, $0(3^{-2k})$, it was pointed out that it is only for small values of k that this factor is significant. This is the effect of the error factor for the first k (k = 1).

We start with X, a normalized (in the ordinary sense) ternary number, i.e., a number with a non-zero digit immediately to the right of the ternary point. This defines two initial ranges (.TT ··· T, .T1 ··· 1) and (.1T ··· T, .11 ··· 1) such that $X \in ([-\frac{1}{2}, -\frac{1}{6}] \cup [\frac{1}{6}, \frac{1}{2}])$.

The initialization step is $x_1 = x_0 \cdot d_0$ where $x_0 = X$ and

$$d_0 = \begin{cases} -4 & \text{for } -\frac{1}{2} \leq x_0 < -\frac{1}{3} \\ -2 & \text{for } -\frac{1}{3} \leq x_0 \leq -\frac{1}{6} \\ 2 & \text{for } \frac{1}{6} \leq x_0 \leq \frac{1}{3} \\ 4 & \text{for } \frac{1}{3} < x_0 \leq \frac{1}{2} \end{cases}$$

which means that x_1 is in a range which is symmetric around 1: $x_1 \in (\frac{2}{3}, \frac{4}{3})$.

Since in the non-redundant algorithm, we do not proceed to the k+1-th digit until the k-th digit is equal to zero (and one repetition is sufficient), the obvious initial range is usable and it is not necessary to have an initialization step.

The algorithm was also developed by Barak [5] and the proof of convergence is given in [5].

ALGORITHM 2: TERNARY DIVISION USING SEVEN INTERVALS FOR EACH RANGE

1. set $x_0 = X$, determine d_0, initialize $x_1 = x_0 \cdot d_0$, and set k = 1, and $y_1 = 1$;

2. if digit k of x_k is equal to zero then go to 3, otherwise set

$$d_k = 1.0\cdots 0\overline{x_n^{(k)}}\ \overline{x_k^{(k+1)}}\ 0\cdots 0;\ \text{then } x_{k+1} = x_k \cdot d_k \text{ and } y_{k+1} = y_k \cdot d_k;$$

3. set k = k+1; if k = n then we are done and $\frac{Y}{X} = y_{k+1}$, otherwise go to 2.

Statistically, we can show that the percentage of zero digits is approximately 60%, giving this algorithm an approximate shift average of $2\frac{2}{3}$. That is, of the 27 iterations only about 11 of them actually do something. The number of operations actually performed is approximately 18.

When the original range is divided into seven intervals, we still check the k-th digit (at step k) and only if it is non-zero do we perform some operation(s). It is possible to generalize Algorithm 2 such that this check is eliminated.

We eliminate the check and continue to define d_k as we did for the algorithm for seven intervals. Thus the original range is divided into nine intervals, one for each of the possible combinations of two trits.

ALGORITHM 3: TERNARY DIVISION USING NINE INTERVALS FOR EACH RANGE

1. set $x_0 = X$, determine d_0, initialize $x_1 = x_0 \cdot d_0$, and set k = 1, and $y_1 = 1$;
2. set $d_k = 1.0\cdots 0\overline{x_k^{(k)}}\,\overline{x_k^{(k+1)}}0\cdots 0$ then $x_{k+1} = x_k \cdot d_k$ and $y_{k+1} = y_k \cdot d_k$;
3. set k = k+1; if k = n then we are done and $\frac{Y}{X} = y_{k+1}$, otherwise go to 2.

This algorithm is essentially an error-correcting version of Algorithm 1. That is, by always mapping successfully one level down, we always generate a zero digit for digit k. Only when a carry is propagated will digit k+1 not be equal to zero. Since we check digit k+1 in any case at the next iteration, it is not necessary to perform the extra check as we did at step 3 of Algorithm 1. This is also the reason that even though we try to get two non-zero digits, we cannot check the digits in groups of two. Statistically, we found that the number of operations is approximately equal to 18. The number of iterations which actually do something is also approximately equal to 18.

When we use simple constants of two ternary digit it is possible to map successfully two levels down. Now we modify the algorithm such that we use not-so-simple constants of three ternary digits and we aim to map successfully three levels down.

ALGORITHM 4: TERNARY DIVISION USING NINETEEN INTERVALS FOR EACH RANGE

1. set $x_0 = X$, determine d_0, initialize $x_1 = x_0 \cdot d_0$, set k = 1 and $y_1 = 1$;
2. set $d_k = 1.0\cdots 0\overline{x_k^{(k)}}\,\overline{x_k^{(k+1)}}\,\overline{x_k^{(k+2)}}0\cdots 0$; then $x_{k+1} = x_k \cdot d_k$ and $y_{k+1} = y_k \cdot d_k$;
3. set k = k+1, if k = n then we are done and $\frac{Y}{X} = y_{k+1}$, otherwise go to 2.

It turns out that even when three digits constants are used, it is not always possible to map successfully three levels down. The percentage of numbers in the "original" range which do map successfully three levels down does however increase as k increases. However, the amount of bookkeeping increases as the number intervals increases. Statistically, the number of iterations where something actually happens is approximately equal to $8\frac{1}{2}$ and the corresponding number of operations is slightly less than 20.

V. EFFICIENCY OF BINARY MULTIPLICATION ALGORITHMS

The normalization process used for division has two important features: 1) an operand is normalized to the constant one, and 2) the process involves continued product which is, in effect, a shift followed by an addition/subtraction operation.

The normalization process used for multiplication also results in a 'shift' followed by an addition/subtraction operation but it is different from the normalization process used for division in that 1) the operand is normalized to zero, and 2) the process involves continued summation. Another significant difference is that the second operand must be held in a separate high-speed register throughout the entire process.

The algorithm, taken from [2], for the multiplication of binary floating point numbers involves an implicit recoding of the multiplier into a redundant form. Although the form is not necessarily canonical, the resultant shift average is very close to 3.

We wish to form the product of two numbers X and Y:

$$p = X \cdot Y = x \cdot 2^{\alpha} \cdot y \cdot 2^{\beta} = x \cdot y \cdot 2^{(\alpha+\beta)} = p \cdot 2^{(\alpha+\beta)}$$

where $p = x \cdot y$, $p \in [\frac{1}{4},1)$ and $x, y \in [\frac{1}{2},1)$, α, β are integers. The sum $(\alpha+\beta)$ is a simple addition.

We rewrite $p = x \cdot y$:

$$p = x \cdot y = y(x - \sum_{k=1}^{n} m_k + \sum_{k=1}^{n} m_k)$$

where $m_k = s_k \, 2^{-k}$, $s_k \in \{-1,0,1\}$. The object of the normalization is to get $x - \sum_{k=1}^{n} m_k = 0$ such that

$$p = y \cdot \sum_{k=1}^{n} m_k \quad .$$

Two simple recursion relations follow from this (i.e. the operand is normalized and the product is formed simultaneously and independently):

$$x_{k+1} = x_k - m_k; \text{ and } \quad p_{k+1} = p_k + y \cdot m_k \quad .$$

Again, the m_k are defined in terms of another constant, s_k:

$$m_k = \frac{1}{2} s_k \, 2^{-k} \quad \text{where}$$

$$s_k = \begin{cases} -1 & x_k < -\frac{3}{8} 2^{-k} \\ 0 & \text{otherwise} \\ 1 & x_k \geq \frac{3}{8} 2^{-k} \end{cases}$$

This algorithm is based on ranges of numbers symmetric around zero and since normalized binary floating point numbers are in the interval $[\frac{1}{2}, 1)$, we must define an initialization step.

VI. EFFICIENCY OF TERNARY MULTIPLICATION ALGORITHMS

The algorithm for multiplication of ternary numbers (taken from Barak [4]) is simpler than the equivalent binary algorithm. The recursion relations are the same:

$$p = X \cdot Y = x \cdot 3^{\alpha} \cdot y \cdot 3^{\beta} = x \cdot y \cdot 3^{(\alpha+\beta)} = p \cdot 3^{(\alpha+\beta)}$$

where p, x, α, β are integers.

Since $p = x \cdot y = y(x - \sum_{k=1}^{n} m_k + \sum_{k=1}^{n} m_k)$, we get

$$x - \sum_{k=1}^{n} m_k = 0; \quad \text{and } p = y \cdot \sum_{k=1}^{n} m_k .$$

So we can define $x_{k+1} = x_k - m_k$ and $p_{k+1} = p_k + y \cdot m_k$, $m_k = s_k \, 3^{-k}$ where $p_0 = 0$ and

$$s_k = \begin{cases} -1 & x_k < -\frac{1}{2} 3^{-k} \\ 0 & \text{otherwise} \\ 1 & x_k > \frac{1}{2} 3^{-k}. \end{cases}$$

In other words, m_k is a ternary number with at most one non-zero digit and this is in the k-th position after the ternary point; s_k is the k-th digit of x_k. y must be available at every step.

At step k, digit k of the partially normalized multiplier, x_k, is checked. If it is non-zero, y is shifted k places to the right and added to or subtracted from the partially formed product, p_k, according to the sign of digit k of x_k.

ALGORITHM 5: TERNARY MULTIPLICATION USING THREE INTERVALS FOR EACH RANGE

1. set $x_1 = X$, $k = 1$, $p_1 = 0$.
2. set $m_k = 0.0\cdots0\overline{x_k^{(k)}}0\cdots0$ and perform

   $p_{k+1} = p_k + y \cdot m_k$, and $x_{k+1} = x_k - m_k$.
3. set $k = k+1$; if $k = n$ then we are done and $p = p_{k+1}$ otherwise, go to step 2.

It is also possible to define an algorithm for multiplication which uses more than three intervals for each range.

ALGORITHM 6: TERNARY MULTIPLICATION USING SEVEN INTERVALS FOR EACH RANGE

1. set $x_1 = X$, $k = 1$, $p_1 = 0$.
2. set $m_k = 0.0\cdots0\overline{x_k^{(k)} \; x_k^{(k+1)}}\,0\cdots0$ and perform

   $p_{k+1} = p_k + y \cdot m_k$; and $x_{k+1} = x_k - m_k$.
3. set $k = k+1$; if $k = n$ then we are done and $p = p_{k+1}$ otherwise, go to step 2.

Statistically, we have found that, using Algorithm 5, there are 18 'useful' iterations. Since one operation is performed at each 'useful' iteration, about 18 operations are performed and the shift average is about $\frac{3}{2}$. Using Algorithm 6, there are only about 11 'useful' iterations but the total number of operations performed is also approximately equal to 18. The shift average for Algorithm 6 is about $2\frac{2}{3}$.

It is possible to present an algorithm constructed in the same manner which uses 9 intervals for each range. That is, we divide each of the three intervals of Algorithm 5 into three intervals each, giving 9 intervals in all. (In order to construct Algorithm 6 from Algorithm 5, we divided only the two outer intervals into three intervals each and left the middle interval as it was). The advantage of doing this with respect to division is that it eliminates the check for overflow. Since overflow occurs when a carry is propagated, and since there is no possibility of a carry occurring with respect to multiplication, there is no conceptual advantage in defining an equivalent 9 interval algorithm. Be that as it may, we collected statistics on a 9 interval algorithm anyway and found that it needed about 18 'useful' iterations and performed about 18 operations, giving a shift average of about $\frac{3}{2}$.

REFERENCES

1. Robertson, J.E.: Introduction to Digital Computer Arithmetic, Digital Computer Laboratory, University of Illinois, File No. 599, June 5, 1964.

2. De Lugish, B.G.: A Class of Algorithms for Automatic Evaluation of Certain Elementary Functions in a Binary Computer, Department of Computer Science, University of Illinois, Report 399, June 1, 1970.

3. Frieder, G. and Luk, C.: Algorithms for Binary Coded Balanced and Ordinary Ternary Operations, IEEE Trans. on Electronic Computers, to appear.

4. Barak, A.: Basic Algorithms for Ternary Arithmetic on a Binary Computer, Department of Computer Science, Hebrew University, Jerusalem, Report 9, March 1, 1974.

5. Barak, A.: An Efficient Algorithm for Computing Natural Logarithms Using Ternary Arithmetic, Department of Computer Science, Hebrew University, Jerusalem, Report 11, July 1, 1974.

HYBRID LOGIC [+]

(A Fast Ternary Adder)

Claudio Moraga [++]
Universität Dortmund
West Germany

Abstract: The concept of Hybrid Logic is introduced, as opposed to that of one-type-only Logic. A fast Ternary Adder is designed both with Hybrid Logic and other ternary logics, enhancing the benefits of the proposed Logic concept.

Index Terms: Hybrid Logic, Ternary Threshold Logic, Ternary Adders.

1. Introduction:

Often it ist found in the literature a defense of one-type-only design over other possibilities. This paper argues that it does not exist a best logic for any design to be relized, but a best logic design for a given problem, using the most suitable logic for each possible part of the problem. For this design philosophy, the mane "Hybrid Logic" has been chosen.

For the purpose of enlighting this idea, a ternary adder of the Killburn type [1] has been designed, as it is in a Killburn adder where one must accept, there are parts of a design where clearly "better-than-gates" implementations are optimum.

First it is shown how easily a Killburn model may be extended to the ternary case, but no further emphasis is made on the speed feature of the carry propagation path, as this is assumed to be well known. Then, the same idea of adapting the logic design to the best possible implementation, according to the characteristics of the problem, is introduced in the logic design of the adding per-digit modules, showing the improvements in overal speed.

+ This research was begun under the sponsorship of the Research Commission, Universidad Santa Maria, Chile and ended under a Fellowship from the Alexander von Humbold Foundation.

++ Universität Dortmund, Abteilung Informatik, Bundesrepublick Deutschland, on leave from:

Universidad Santa Maria, Departamento de Computación, Valparaiso, Chile.

2. Notation and Definitions:

Def. 1: Let V = (0,1,2) be the set of true values for a ternary variable. $\forall i\ X_i \in V$, let $\underline{X} = (X_1, X_2, \ldots X_n)$ be a ternary vector. $\forall F(\underline{X}) \ni F(\underline{X}) \in V$, $F(\underline{X})$ is a ternary function. A ternary function sets a partition over V^n, defining three blocks, namely F_0, F_1 and F_2 which are mapped onto 0,1 and 2 respectively.

Def. 2: $F(\underline{X})$ is a ternary Threshold function if and only if $\exists$ $(w_1, w_2, \ldots w_n, H, L)$, with $w_i, H, L \in \mathbb{R}$, $H \geqslant L$, which satisfy the following equations:

$$\begin{aligned} L > \underline{X}.\underline{w} \quad &<=> \quad F(\underline{X}) = 0 \\ H > \underline{X}.\underline{w} > L \quad &<=> \quad F(\underline{X}) = 1 \\ \underline{X}.\underline{w} > H \quad &<=> \quad F(\underline{X}) = 2 \end{aligned} \qquad (1)$$

From the geometrical point of view, eq.(1) states the existence of two (n-1) - dimensional parallel hyperplanes in V^n, such that they separate F_0 from F_1 and F_1 from F_2 in that order.

Def. 3: $F(\underline{X})$ is a Multithreshold Periodic ternary function [2], if and only if $\exists$ $(w_1, w_2, \ldots w_n, k)$, with $w_i, k \in \mathbb{R}$; $|w_i|$, $|k| < 1$, which satisfy the following equations:

$$\begin{aligned} 0 < (\underline{X}.\underline{w} + k) \bmod 1 < 1/3 \quad &<=> \quad F(\underline{X}) = 0 \\ 1/3 < (\underline{X}.\underline{w} + k) \bmod 1 < 2/3 \quad &<=> \quad F(\underline{X}) = 1 \\ 2/3 < (\underline{X}.\underline{w} + k) \bmod 1 < 1 \quad &<=> \quad F(\underline{X}) = 2 \end{aligned} \qquad (2)$$

From the geometrical point of view, $F(\underline{X})$ is Multithreshold Periodic, if and only if there exist a set of parallel equidistant (n-1) - dimensional hyperplanes which partitions

V^n separating succesive subsets of F_0,F_1 and F_2 in that order.

Def. 4: If $F(\underline{X})$ is a Threshold function, it may be represented as follows:

$$F(\underline{X}) : (\underline{w},H,L) \qquad (3)$$

If $F(\underline{X})$ is a Multithreshold Periodic function, it may be represented as follows:

$$F(\underline{X}) : (\underline{w},k) \qquad (4)$$

3. Fast Adder Design:

In a first design, let ternary digits be chosen from V =(0,1,2), thus the carry is binary [3], as may be seen in figs. 1 and 2. This allows a straight forward application of a Killburn carry network [1] for fast behavior of the overall machine.

The design is based on per-digit adding modules. There are three kind of modules: a Half-Adder, HA, for the least significant digits (order zero); a Full-Adder FA, for the most significant digits (order n) and n-1 Gate-Adders, GA, for the remaining digits.

A GA has the same inputs and S_i output as a FA. However, instead of a C_i carry output, it has a P_i and Q_i outputs which are independent from C_{i-1}, as shown in fig. 3. These P_i and Q_i signals gate on and off the carry propagation path.

Comparing the carry output of a FA (fig.2) with the P_i and Q_i functions (fig. 3), it may be seen that:

$$\overline{P}_i\ Q_i \quad => \quad C_i = C_{i-1} \qquad (5)$$

$$P_i\ Q_i \quad => \quad C_i = 0 \qquad (6)$$

$$\overline{Q}_i \quad => \quad C_i = 1 \qquad (7)$$

These equations define the Killburn carry propagation network and show how simply a ternary version is possible.

The logic block diagram of the machine is shown in fig. 4.

4. Speed Analysis:

In the block diagram of the Adder, it may be seen that the overall operating time is that required by the carry generation in the HA plus the operating time of slowest GA or FA. (The carry propagation time may be deleted as it is much smaller than that of the fastest module).

For the speed analysis, let it be assumed that d is the delay time (operating time) of one single Max, Min or Unary function Gate; let 3d be the delay time of a threshold gate, and let 4d be the delay time of a multithreshold gate. These delay times are obviously an oversimplified, reverse-biased relative assumption, and they should be considered like so, yet they allow to visualize the differences among the logic designs which will be discussed below.

4.1 Logic Design by Sum of Products of Unary Functions:

In this case, (see fig. 5), a HA requires four level logic for the S_0 trit and three level logic for the C_0 bit. This means that a HA has an operating time of 4d (3d for C_0). A direct FA design becomes very complex but a FA may be obtained with three cascaded HAs [3], as shown in fig. 6. Then, for the worst case, a FA requires 12d.

Since for a GA no C_i bit is required, but only the S_i trit, only two cascaded HA are needed, as may be seen in fig. 6. The operating time of a GA is then 8d.

Thus, with this type of design, the complete Adder would have a 15d operating time.

4.2 Logic Design by Cascaded Threshold Logic:

A cascaded threshold design has been suggested [7] for adding machines. The theoretic design is appealing, but weights increase as 3^n , and that poses severe constraints in tolerance for the electronic implementation [6], besides, it is also of the ripple-through carry type.

However, the cascaded threshold design seems convenient for HAs. GAs and FAs, as it only requires two level threshold logic, with a maximum weight of 3, which means a 6d operating time as may be seen in fig. 7. Note that only 3d is required for C_0.

Thus, with this type of design, the complet Adder would have a 9d operating time.

4.3 Logic Design by Hybrid Logic:

Speed of the Adder may be further improved by using Hybrid Logic. In this case, today, that means using threshold and multithreshold periodic design for the individual modules, as the required digits exhibit these characteristics.

In a HA (fig. 1), it may easily be seen that S_0 is a multithreshold periodic function, and can reliably be implemented with one single gate [3,4,5]. C_0 is seen to be a two-valued threshold

function of ternary variables and can also be implemented with one single gate. Thus following eq. (3) and eq. (4):

$$S_0 : (1/3, 1/3, 1/6) \quad (8)$$

$$C_0 : (1, 1, \infty, 5/2) \quad (9)$$

In a FA (fig. 2), it is possible to see that S_n is also a multithreshold periodic function, and C_n is again a two-valued threshold function; so:

$$S_n : (1/3, 1/3, 1/3, 1/6) \quad (1o)$$

$$C_n : (1, 1, 1, \infty, 5/2) \quad (11)$$

In a GA, the S_i trit may be obtained as in the FA by means of a multithreshold periodic gate. Both P_i and Q_i are two valued threshold functions, as may be seen in fig. 3. Then:

$$P_i : (1, 1, \infty, 5/2) \quad (12)$$

$$Q_i : (1, 1, \infty, 3/2) \quad (13)$$

Thus, in any module the S trit generation would require 4d and the C,P or Q bit generation, 3d. Consequently, the complete Adder would require only a 7d operating time.

5. Implementations of the Hybrid Logic Design:

Good reliable circuits for multithreshold logic have been reported [4,5] as well as circuits for threshold logic [4,5,6].These circuits allow a convenient implementation of the adding modules as defined in eqs. 8 through 13. The carry propagation network may be implemented with integrated transistors or integrated switches.

6. Symmetric Arithmetic:

When (-1,0,1) is chosen as the set of true values for the ternary variables instead of (0,1,2), a symmetric number representation is

obtained. This number representation provides attractive properties for arithmetic processing. (Direct representation of both positive and negative numbers, implicit sign in arithmetic operations and equal round-off and truncation errors.)

Should a symmetric Adder be desired, the above discussed design idea is also possible. In this case C_i is no longer a bit, but a trit; however, a special fast carry propagation network has been suggested [8] for three-valued carry. Gate signals to control this path require more complex GAs, but the overall design philosophy may be preserved.

(See Appendix 1)

7. Conclusions:

Table 1: Summary

Design type	Nr. of levels	Relative Operating Time				
		HA	C_0	GA	FA	Adder
unary functions	4	4d	3d	8d	12d	15d
cascaded threshold	2	6d	3d	6d	6d	9d
Hybrid	1	4d	3d	4d	4d	7d

It has been shown that a ternary Killburn Adder is simple to design. But more important, it has been shown that a Hybrid logic design -(in this case: threshold - multithreshold periodic - switch logic)- provides a better solution than a "uniform" logic design. (see Table 1).

It is concluded then, that more multiple-valued logics should be studied an developed, to enlarge the scope from where a more powerful Hybrid Logic may rise.

8. Bibliography:

(1) Killburn et al.: "A Parallel Arithmetic Unit using a Saturated Transistor Fast Carry Circuit". Proc. IEE, 107-B, (11), 573-584, (1960)

(2) Gutiérrez J. and Moraga C.: "Multithreshold Periodic Ternary Threshold Logic", Proc. 1974 international Symposium on Multiple-valued Logic, W.Va., USA, 413-422 (1974)

(3) Mine et al.: "Ternary four Arithmetic Operations" Trans. IECE Japan, 54-C, (1), 66-73,(1971)

(4) Druzeta A. and Sedra A.: "Multithreshold Circuits in the design of Multistate Storage Elements" Proc. 1973 International Symposium on Multiple-valued Logic, Toronto, Canada, 49-58

(5) Druzeta A., Vranesic Z. and Sedra A.: "Applications of Multithreshold Elements in the Realization of Many-valued Logic Networks", IEEE Tr C-23, (11) 1194-1198, (1974)

(6) Bittner A.: " Ternary Implementation of an Automatic Position Control System" E.E. Thesis, Universidad Santa Maria, Valparaiso, Chile, (1974), (in Spanish)

(7) Merrill R.: "Some properties of Ternary Threshold Logic", IEEE Tr. EC-13,(5), 632-635, (1964)

(8) Arango H. and Santos J.: "A Fast Carry-propagation Circuit for Base 3 Signed non Redundant Arithmetic", IEEE Tr. EC-15, (2), 254-255, (1966)

Appendix 1

Symmetric Adder

In a Symmetric Adder, the Carry is ternary, as may be seen in fig. A1, but may well be decomposed into two carry bits, namely $C'_i \in (-1,0)$ and $C''_i \in (0,1)$ as shown in fig. A1, that satisfy the following equation:

$$C'_i \times C''_i = C_i \qquad (A1)$$

($\times$ stands for arithmetic addition)

Four Gate (binary) signals may be used to control a double carry-path. Let these signals be M,N,P and Q, as shown in fig. A2 and fig. A3.

The double carry-path equations are:

$\overline{M}.\overline{N} = C'_i = -1$ (A2)

$\overline{M}.N = C'_i = C'_{i-1}$ (A3)

$M = C'_i = 0$ (A4)

$P.Q = C''_i = 1$ (A5)

$\overline{P}.Q = C''_i = C''_{i-1}$ (A6)

$\overline{Q} = C''_i = 0$ (A7)

The Symmetric Adder requires a modified HA to give C'_0 and C''_0, but these are simple two-valued threshold functions. Also the GAs become more complex, as they must provide the four Gate signals and their binary complements, but these are also simple two-valued threshold functions.

The S_i trits are once more multithreshold periodic functions of four inputs: X_i, Y_i, C'_{i-1} and C''_{i-1}. It is to be noticed, that at the input stage of the multithreshold gate addition of C'_{i-1} and C''_{i-1} takes place thus restablishing the C_{i-1} trit. Therefore S_i continues to be a 3-place function, i.e: $S_i = S_i\ (X_i, Y_i, C_{i-1})$

It may be clearly seen that in this case a Hybrid logic design gives also the best solution. Besides, the Hybrid concept becomes stronger, as here explicitly involves: multithreshold periodic ternary logic, threshold two-valued logic of ternary variables, and duo binary (switch) logic.

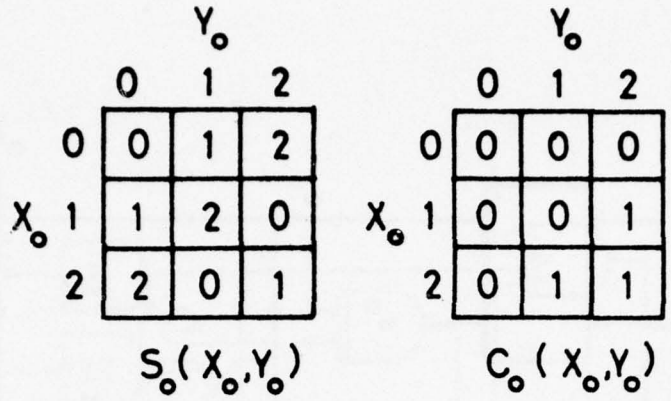


Fig. 1: Ternary Half Adder

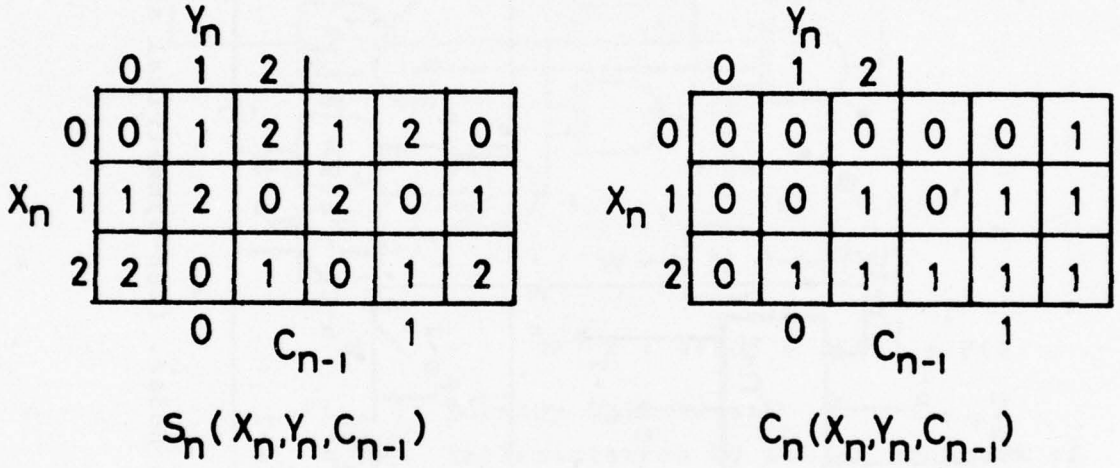


Fig.2: Ternary Full Adder

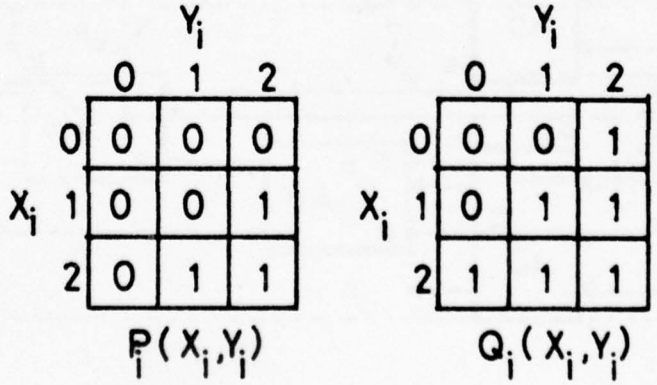


Fig. 3: Path control signals

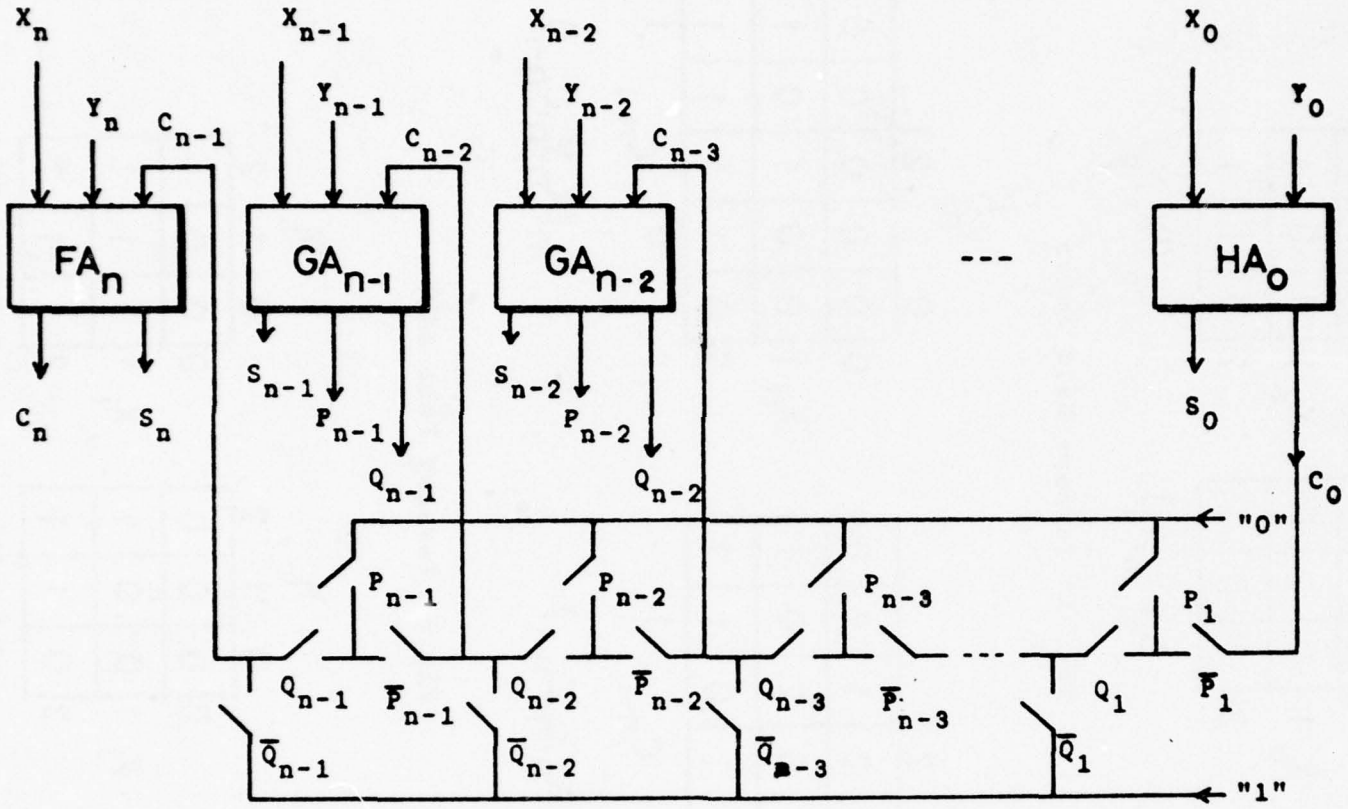
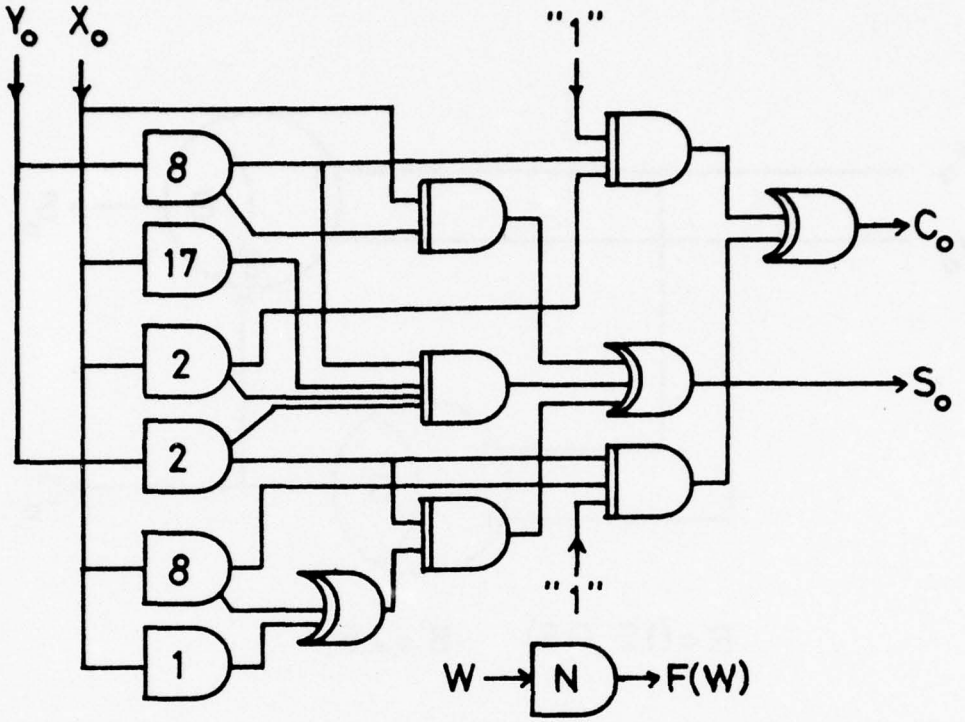


Fig. 4: Killburn Adder. (non-Symmetrical arithmetic)



$$N = 9F(0) + 3F(1) + F(2)$$

Fig. 5: Ternary Half Adder
Implementation by Sum of Products of
unary functions

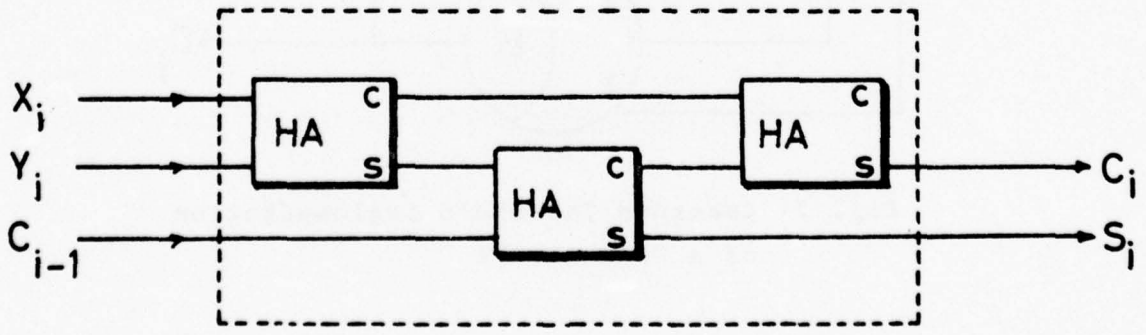


Fig. 6: HA implementation of a Ternary FA

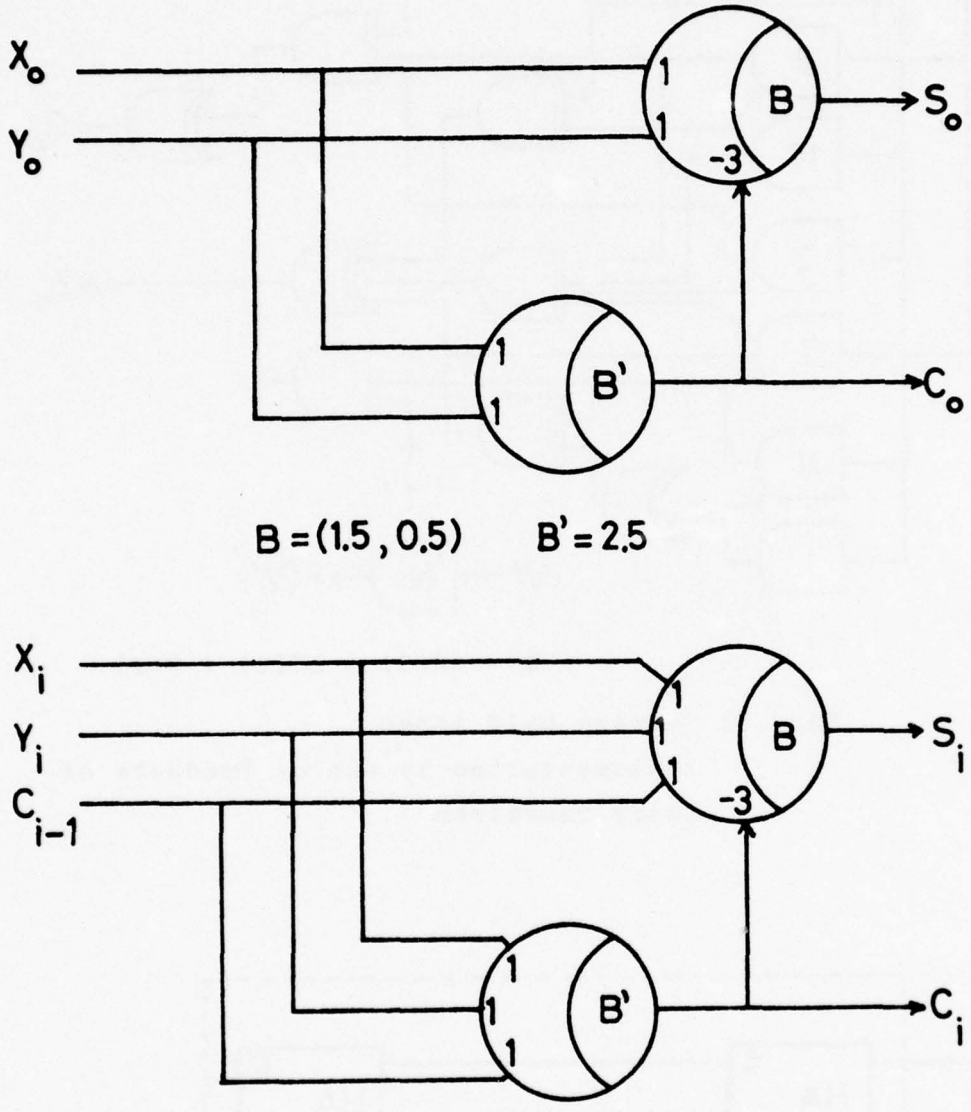
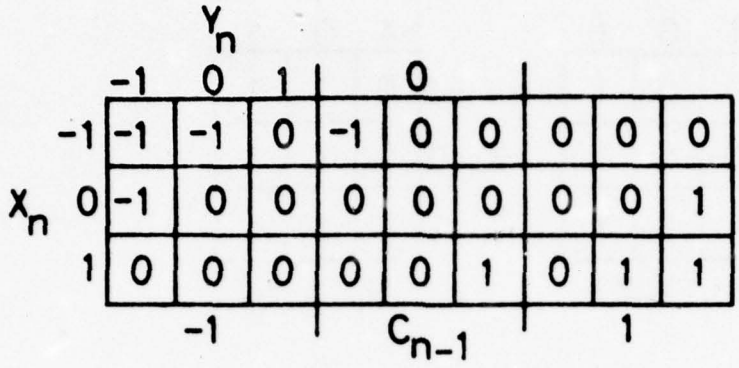
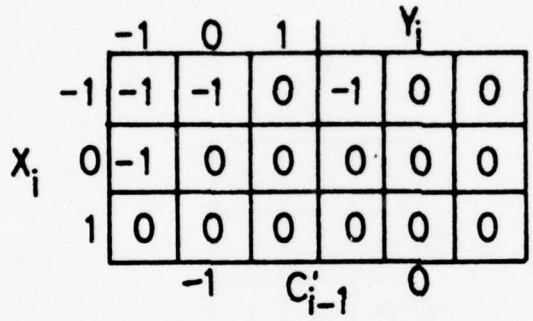


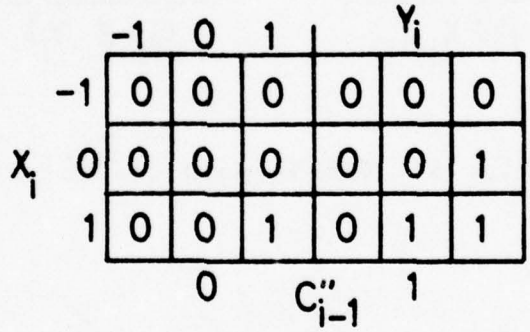
Fig. 7: Cascaded Threshold implementation of a HA and a FA



(a)



(b)



(c)

(a) FA : $C_n(X_n, Y_n, C_{n-1})$

(b) GA : $C'_i(X_i, Y_i, C'_{i-1})$

(c) GA : $C''_i(X_i, Y_i, C''_{i-1})$

Fig. A1: Symmetric ternary adder

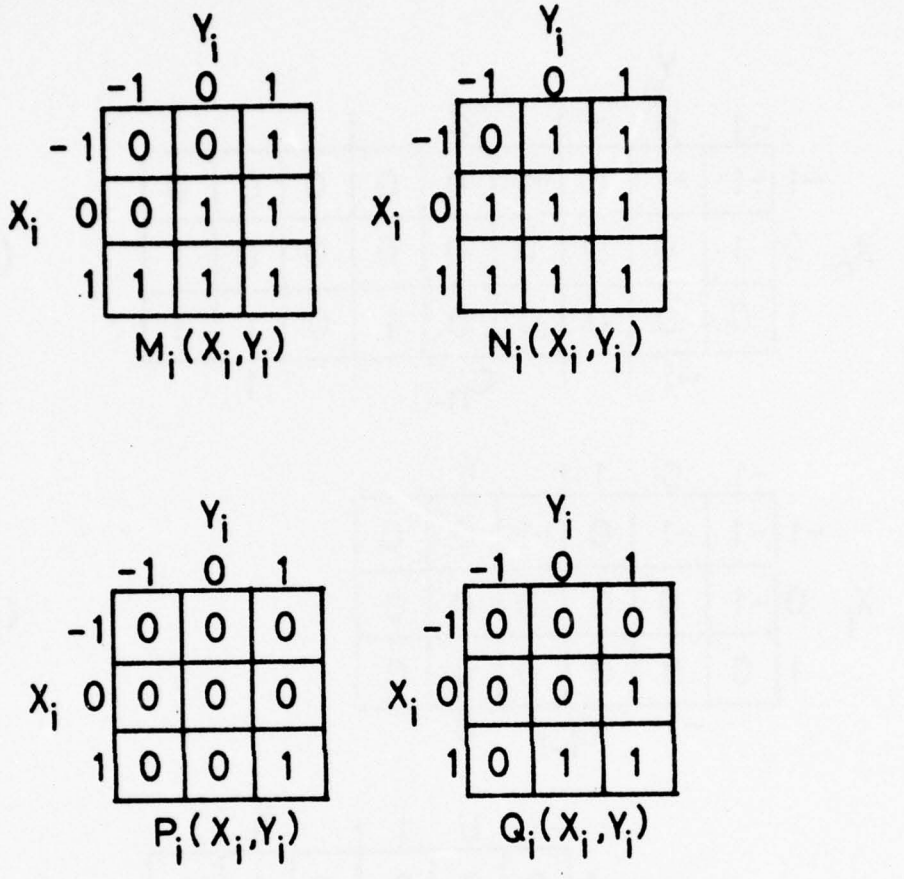


Fig. A2: Double-path control signals

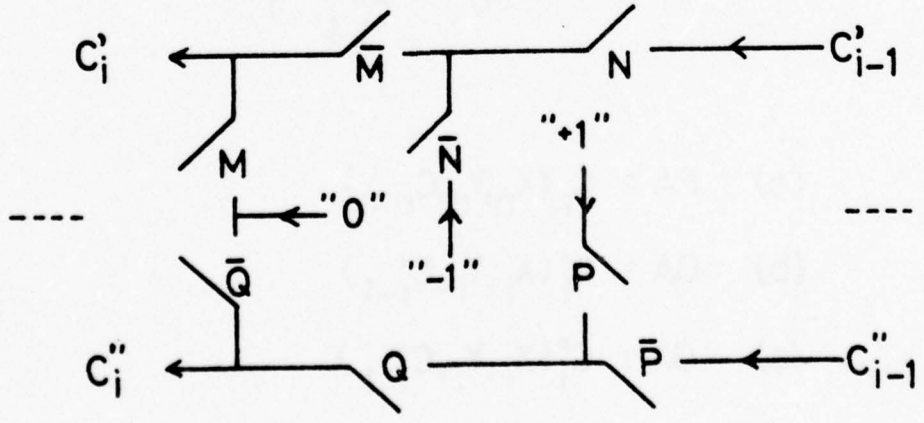


Fig. A3: Gated duo-binary carry double-path

A DESIGN TECHNIQUE FOR AN INTEGRABLE TERNARY ARITHMETIC UNIT

H.T. Mouftah and I.B. Jordan
Département de Génie Electrique
Université Laval
Québec, Canada

ABSTRACT

A design technique for a complete integrable ternary arithmetic unit is presented. A ternary algebra is given and a special method of simplification is described. COS/MOS integrated circuits are applied to realize the basic ternary functions. A design of a ternary full adder is developed. Ternary storage elements and shift registers are presented to design a ternary arithmetic unit.

1. INTRODUCTION

Three-valued logic systems are not yet sufficiently advanced to be as widely used as the two-valued logic systems. This is due to the lack of a suitable ternary memory element as well as a complete integrable ternary circuit at a reasonable cost. Attempts have been made to solve this problem by the application of the Complementary-Symmetry Metal Oxide Semiconductor (COS/MOS) [1-3]. In the present paper, which is a continuation of these works, a design technique for a complete integrable ternary arithmetic unit is presented. The algebra includes ternary operators previously given by Halpern and Yoeli [4].

2. THE ALGEBRA

The three voltage levels of ternary logic circuits will be presented here by 0, 1 and 2: 0 for the low (-4V), 1 the intermediate (zero potential) and 2 the high level (+4V).

Let $L = \{0,1,2\}$. In L a set of operators are defined. The simple ternary inverter (STI), positive ternary inverter (PTI) and negative ternary inverter (NTI) are considered as basic unary operators. These are presented by Table I and the following equations

$$STI \equiv \overline{x^1} = 2 - x$$

$$PTI, NTI \equiv \overline{x^i} = \begin{cases} i & \text{if } x \neq i \\ 2 - i & \text{if } x = i \end{cases}$$

where i can be 2 or 0.
The minus sign in the above two equations represents arithmetic subtraction.

The operation of addition (+) and multiplication (·) on L, which can be called ternary OR (TOR) and ternary AND (TAND) respectively, represent two multiple entry operators. These are given in Table II and the following equations

$$TOR \equiv MAX(x,y) = x + y$$

$$TAND \equiv MIN(x,y) = x \cdot y$$

For any $x,y,z \in L$, and the constants 0 and 2, the following theorems hold.

Theorem 1: Identity elements

a) $x + 0 = x$ b) $x \cdot 2 = x$

Theorem 2:

a) $x + 2 = 2$ b) $x \cdot 0 = 0$

Theorem 3: Idempotent

a) $x + x = x$ b) $x \cdot x = x$

Theorem 4: Commutative

a) $x + y = y + x$ b) $x \cdot y = y \cdot x$

Theorem 5: Associative

a) $(x+y) + z = x + (y+z)$ b) $x \cdot (y \cdot z) = (x \cdot y) \cdot z$

Theorem 6: Absorption

a) $x + (x \cdot y) = x$ b) $x \cdot (x+y) = x$

Theorem 7: Distributive

a) $x+(y \cdot z) = (x+y) \cdot (x+z)$ b) $x \cdot (y+z) = (x \cdot y) + (x \cdot z)$

Theorem 8: De Morgan's

1.a) $\overline{(x+y)^0} = \overline{x^0} \cdot \overline{y^0}$ 1.b) $\overline{(x \cdot y)^0} = \overline{x^0} + \overline{y^0}$

2.a) $\overline{(x+y)^1} = \overline{x^1} \cdot \overline{y^1}$ 2.b) $\overline{(x \cdot y)^1} = \overline{x^1} + \overline{y^1}$

3.a) $\overline{(x+y)^2} = \overline{x^2} \cdot \overline{y^2}$ 3.b) $\overline{(x \cdot y)^2} = \overline{x^2} + \overline{y^2}$

Theorem 9:

a) $x + \overline{x^2} = 2$ b) $x \cdot \overline{x^0} = 0$

Theorem 10:

a) $(x \cdot y) + (x \cdot \overline{y^2}) = x$ b) $(x+y) \cdot (x+\overline{y^0}) = x$

Theorem 11:

a) $x+(\overline{x^2} \cdot y) = x+y$ b) $x \cdot (\overline{x^0}+y) = x \cdot y$

Theorem 12:

a) $(z \cdot x) + (z \cdot \overline{x^2} \cdot y) = (z \cdot x) + (z \cdot y)$ b) $(z+x) \cdot (z+\overline{x^0}+y) = (z+x) \cdot (z+y)$

Theorem 13 :

a) $(x \cdot y) + (\overline{x^2} \cdot z) + (y \cdot z) = (x \cdot y) + (\overline{x^2} \cdot z)$

b) $(x+y) \cdot (\overline{x^0} + z) \cdot (y+z) = (x+y) \cdot (\overline{x^0} + z)$

Theorem 14 :

a) $\overline{x^0} + \overline{x^1} + \overline{x^2} = \overline{x^2}$

b) $\overline{x^0} \cdot \overline{x^1} \cdot \overline{x^2} = \overline{x^0}$

a-1) $\overline{x^0} + \overline{x^1} = \overline{x^1}$

b-1) $\overline{x^0} \cdot \overline{x^1} = \overline{x^0}$

a-2) $\overline{x^2} + \overline{x^1} = \overline{x^2}$

b-2) $\overline{x^2} \cdot \overline{x^1} = \overline{x^1}$

Theorem 15 :

$$\overline{(\overline{x^1})^1} = x$$

Theorem 16 :

$$\overline{((\overline{x^0})^i)^i} = \overline{x^0}$$

$$\overline{((\overline{x^2})^i)^i} = \overline{x^2}$$

where $i = 0$, 1 or 2.

Theorem 17 :

$$\overline{(\overline{x^1})^2} = \overline{(\overline{x^0})^1}$$

$$\overline{(\overline{x^1})^0} = \overline{(\overline{x^2})^1}$$

Theorem 18 :

$$\overline{(\overline{x^2})^0} = \overline{(\overline{x^2})^1} = \overline{(\overline{x^2})^2}$$

$$\overline{(\overline{x^0})^0} = \overline{(\overline{x^0})^1} = \overline{(\overline{x^0})^2}$$

Theorem 19 :

1.a) $\overline{x^0} + \overline{(\overline{x^0})^i} = 2$

1.b) $\overline{x^0} \cdot \overline{(\overline{x^0})^i} = 0$

2.a) $\overline{x^2} + \overline{(\overline{x^2})^i} = 2$

2.b) $\overline{x^2} \cdot \overline{(\overline{x^2})^i} = 0$

where $i = 0$, 1 or 2.

Ternary functions of one or more variables may be represented in truth table or map form or algebraically in canonical form as a product-of-sums or as a sum-of-products. An example of representation of a ternary function of two variables in truth table and map

form is given in Table III.

Theorem :

Any ternary function $f(x_1,x_2,\ldots,x_n)$ may be generated from $x_1,x_2,\ldots,x_n$ by means of +,·, unary functions $\overline{x^0},\overline{x^1},\overline{x^2}$ and constant 1. It has been proven [4] that an algebra composed of these operators is functionally complete, and the above theorem holds.

Any ternary function of n variables can be presented by

$$f(x_1,\ldots,x_n) = 2\cdot F_2(x_1,\ldots,x_n) + 1\cdot F_1(x_1,\ldots,x_n) + 0\cdot F_0(x_1,\ldots,x_n)$$

i.e. $f = 2\cdot F_2 + 1\cdot F_1 + 0\cdot F_0$

where F_K equal 2 when the value of the function f equal K, otherwise it will equal 0.

Using theorems 1(b) and 2(b) the function may be presented by

$$f = F_2 + 1\cdot F_1 \quad .$$

3. IMPLEMENTATION OF TERNARY CIRCUITS

The design technique will depend essentially on a minimization method which takes advantage of the technological construction of the circuits realizing the operators of the algebra presented above.

A positively inverted input TAND is equivalent to a TAND gate with a PTI circuit inserted in its input. Referring to De Morgan's laws this will be equivalent to a TOR gate with a positively inverted output which can be called positive ternary NOR (or positive TNOR).

$$\overline{x^2}\cdot\overline{y^2} = \overline{(x+y)^2}$$

This is represented in Fig. 4. The same idea holds for the representation of the positively inverted input TOR by the TAND gate with a positively inverted output (positive TNAND).

Similarly, a negatively and simply inverted input TAND (TOR) will be equivalent to a negative and simple TNOR (TNAND) respectively.

$$\overline{x^0}\cdot\overline{y^0} = \overline{(x+y)^0}$$

$$\overline{x^1}\cdot\overline{y^1} = \overline{(x+y)^1}$$

Fig. 1 shows the circuit realization of the STI, PTI and NTI functions constructed by means of COS/MOS integrated circuits. A TNOR and TNAND gates are also given in Fig. 2 and 3 respectively. The operation of these circuits was previously described in detail [1]. These three circuits could be combined in only one circuit [3].

As the inverter circuit of Fig. 1 has three outputs, the TNOR and TNAND circuits of Fig. 2 and 3 have also three outputs each. The outputs A will represent a positive TNOR and positive TNAND respectively. This is equivalent to a TOR (or TAND) gate followed by a

PTI circuit. The outputs B will represent a negative TNOR and negative TNAND respectively, which is equivalent to a TOR (or TAND) gate followed by a NTI circuit. Similarly, outputs D will represent a simple TNOR and simple TNAND doing the function of a TOR (or TAND) followed by a STI circuit. Performances of these operators are shown in Table IV.

The ternary inverter circuit of Fig. 1 can be represented by a closed box having three outputs P, S and N replacing the A, D and B on the schematic diagram. This ternary inverter will be able to drive three different ternary inverters having nine outputs, each can drive a third level of ternary inverters to supply 27 outputs. But the third level is not needed since its outputs will be the same as the first one (theorem 16). Also the ternary inverter (4) in the second level can be omitted because its outputs could be taken from ternary inverters (2) and (3) (theorem 17 and 18) or from the variable itself (theorem 15). Finally the six outputs of ternary inverters (2) and (3) can be reduced to only two outputs, one for each inverter (theorem 18). This can be shown clearly in Fig. 5. Therefore 5 outputs could be taken from three ternary inverters connected as shown in the figure.

Methods of minimization of combinational ternary switching functions, previously described by several authors [5-9] can be applied to the algebra presented above. Most of these methods were applied to algebras composed of the MAX and MIN operations and functions defined by

$$x^i = \begin{cases} 0 & \text{if } x \neq i \\ 2 & \text{if } x = i \end{cases}$$

where i can be 0, 1 or 2.
To be able to apply these methods the following transformations have to be made

$$x^0 = \overline{x^0}$$

$$x^1 = \overline{(x + \overline{x^1})^2}$$

$$x^2 = \overline{\overline{x^2}^2}$$

Verifications for these transformations can be done by the proof of the following relation

$$x^0 + x^1 + x^2 = 2$$

Proof :

$$x^0 + x^1 + x^2 = \overline{x^0} + \overline{(x + \overline{x^1})^2} + \overline{\overline{x^2}^2}$$

$$= \overline{x^0} + (\overline{x^2} \cdot \overline{\overline{x^1}^2}) + \overline{\overline{x^2}^2} \qquad \text{(theorem 8)}$$

$$= \overline{x^0} + \left[(\overline{x^2} + \overline{\overline{x^2}^2}) \cdot (\overline{\overline{x^1}^2} + \overline{\overline{x^2}^2})\right] \qquad \text{(theorem 7)}$$

$$= \overline{x^0} + \overline{\overline{x^1}^2} + \overline{\overline{x^2}^2} \qquad \text{(theorem 19)}$$

$$= \overline{x^0} + \overline{(\overline{x^1} \cdot \overline{x^2})^2} \qquad \text{(theorem 8)}$$

$$= \overline{x^0} + \overline{\overline{x^1}^2} \qquad \text{(theorem 14)}$$

$$= \overline{x^0} + \overline{\overline{x^0}^1} \qquad \text{(theorem 17)}$$

$$= 2 \qquad \text{(theorem 19)}$$

Therefore functions could be directly written from truth tables as usually. Then algebraic minimization can be followed using the above theorems and taking into consideration the following points:

1. To convert the TAND and TOR to TNAND and TNOR since they are more economical.
2. The simplification has to be oriented to obtain STI, PTI or NTI in the first level if possible.
3. In the presence of a second level, the STI operator in the first level has to be changed into PTI or NTI or into the variable itself. Then a simplification has to be made for obtaining the same operator in the second level.
4. A third level is not allowed because its output is the same as the output of the first one.

4. DESIGN OF A TERNARY FULL ADDER

Consider a stage of a ternary full adder having three inputs x, y, c and two outputs Q and S. x and y are the ternary digits of the two numbers to be added, c is the carry in, Q is the carry out to the next stage and S is the ternary digit of the sum. Fig. 6 shows the sum and carry out to the next stage of a ternary full adder presented on a map form. The sum function can be written directly as

$$S = F_2 + 1 \cdot F_1$$

$$= \Big[\overline{(x+\overline{x^1})^2}\,\overline{\overline{y^2}^2}\,\overline{\overline{c^2}^2} + \overline{\overline{x^2}^2}\,\overline{(y+\overline{y^1})^2}\,\overline{\overline{c^2}^2} + \overline{x^0}\,\overline{y^0}\,\overline{\overline{c^2}^2} + \overline{\overline{x^2}^2}\,\overline{\overline{y^2}^2}\,\overline{(c+\overline{c^1})^2} + \overline{x^0}\,\overline{(y+\overline{y^1})^2}\,\overline{(c+\overline{c^1})^2} +$$

$$+ \overline{(x+\overline{x^1})^2}\,\overline{y^0}\,\overline{(c+\overline{c^1})^2} + \overline{x^0}\,\overline{\overline{y^2}^2}\,\overline{c^0} + \overline{(x+\overline{x^1})^2}\,\overline{(y+\overline{y^1})^2}\,\overline{c^0} + \overline{\overline{x^2}^2}\,\overline{y^0}\,\overline{c^0}\Big] + 1 \cdot \Big[\overline{x^0}\,\overline{\overline{y^2}^2}\,\overline{\overline{c^2}^2} +$$

$$+ \overline{(x+\overline{x^1})^2}\,\overline{(y+\overline{y^1})^2}\,\overline{\overline{c^2}^2} + \overline{\overline{x^2}^2}\,\overline{y^0}\,\overline{\overline{c^2}^2} + \overline{(x+\overline{x^1})^2}\,\overline{\overline{y^2}^2}\,\overline{(c+\overline{c^1})^2} + \overline{\overline{x^2}^2}\,\overline{(y+\overline{y^1})^2}\,\overline{(c+\overline{c^1})^2} +$$

$$+ \overline{x^0}\,\overline{y^0}\,\overline{(c+\overline{c^1})^2} + \overline{\overline{x^2}^2}\,\overline{\overline{y^2}^2}\,\overline{c^0} + \overline{x^0}\,\overline{(y+\overline{y^1})^2}\,\overline{c^0} + \overline{(x+\overline{x^1})^2}\,\overline{y^0}\,\overline{c^0}\Big]$$

and the carry out function as

$$Q = \overline{\overline{x^2}^2}\,\overline{\overline{y^2}^2}\,\overline{\overline{c^2}^2} + 1 \cdot \Big[\overline{\overline{x^2}^2}\,\overline{(y+\overline{y^1})^2}\,\overline{\overline{c^2}^2} + \overline{\overline{x^2}^2}\,\overline{y^0}\,\overline{c^2}^2 + \overline{(x+\overline{x^1})^2}\,\overline{\overline{y^2}^2}\,\overline{\overline{c^2}^2} + \overline{(x+\overline{x^1})^2}\,\overline{(y+\overline{y^1})^2}\,\overline{\overline{c^2}^2} +$$

$$+ \overline{(x+\overline{x^1})^2}\,\overline{y^0}\,\overline{\overline{c^2}^2} + \overline{x^0}\,\overline{\overline{y^2}^2}\,\overline{\overline{c^2}^2} + \overline{x^0}\,\overline{(y+\overline{y^1})^2}\,\overline{c^2}^2 + \overline{\overline{x^2}^2}\,\overline{\overline{y^2}^2}\,\overline{(c+\overline{c^1})^2} + \overline{\overline{x^2}^2}\,\overline{(y+\overline{y^1})^2}\,\overline{(c+\overline{c^1})^2} +$$

$$+ \overline{\overline{x^2}^2}\,\overline{y^0}\,\overline{(c+\overline{c^1})^2} + \overline{(x+\overline{x^1})^2}\,\overline{\overline{y^2}^2}\,\overline{(c+\overline{c^1})^2} + \overline{(x+\overline{x^1})^2}\,\overline{(y+\overline{y^1})^2}\,\overline{(c+\overline{c^1})^2} + \overline{x^0}\,\overline{\overline{y^2}^2}\,\overline{(c+\overline{c^1})^2} +$$

$$+ \overline{\overline{x^2}^2}\,\overline{\overline{y^2}^2}\,\overline{c^0} + \overline{\overline{x^2}^2}\,\overline{(y+\overline{y^1})^2}\,\overline{c^0} + \overline{(x+\overline{x^1})^2}\,\overline{\overline{y^2}^2}\,\overline{c^0}\Big]$$

After some algebraic manipulations the sum and carry out functions will be respectively:

$$S = \overline{(x + \bar{x}^1 + \bar{y}^2 + \bar{c}^2)^2} + \overline{(\bar{x}^2 + y + \bar{y}^1 + \bar{c}^2)^2} + \overline{(x + y + \bar{c}^2)^0} + \overline{(\bar{x}^2 + \bar{y}^2 + c + \bar{c}^1)^2} + \overline{(\bar{\bar{x}^0}^0 + y + \bar{y}^1 + c + \bar{c}^1)^2} +$$

$$+ \overline{(\bar{\bar{y}^0}^0 + x + \bar{x}^1 + c + \bar{c}^1)^2} + \overline{(x + \bar{y}^2 + c)^0} + \overline{(x + \bar{x}^1 + y + \bar{y}^1 + \bar{\bar{c}^2}^2)^2} + \overline{(\bar{x}^2 + y + c)^0} + 1 \cdot \Big[\overline{(\bar{\bar{x}^0}^0 + \bar{y}^2 + \bar{c}^2)^2} +$$

$$+ \overline{(x + \bar{x}^1 + y + \bar{y}^1 + \bar{c}^2)^2} + \overline{(\bar{x}^2 + \bar{\bar{y}^0}^0 + \bar{c}^2)^2} + \overline{(x + \bar{x}^1 + \bar{y}^2 + c + \bar{c}^1)^2} + \overline{(\bar{x}^2 + y + \bar{y}^1 + c + \bar{c}^1)^2} +$$

$$+ \overline{(\bar{\bar{x}^0}^0 + \bar{\bar{y}^0}^0 + c + \bar{c}^1)^2} + \overline{(\bar{x}^2 + \bar{y}^2 + \bar{\bar{c}^0}^0)^2} + \overline{(\bar{\bar{x}^0}^0 + y + \bar{y}^1 + \bar{\bar{c}^0}^0)^2} + \overline{(x + \bar{x}^1 + \bar{\bar{y}^0}^0 + \bar{\bar{c}^0}^0)^2}\Big]$$

$$Q = \overline{(\bar{x}^2 + \bar{y}^2 + \bar{c}^2)^2} + 1 \cdot \Big[\overline{(\bar{x}^2 + y + \bar{c}^2)^2} + \overline{(x + \bar{x}^1 + \bar{c}^2)^2} + \overline{(\bar{\bar{x}^0}^2 + \bar{y}^1 + \bar{c}^2)^2} + \overline{(\bar{x}^2 + c + \bar{c}^1)^2} +$$

$$+ \overline{(x + \bar{x}^1 + \bar{y}^1 + c + \bar{c}^1)^2} + \overline{(\bar{\bar{x}^0}^2 + \bar{y}^2 + c + \bar{c}^1)^2} + \overline{(\bar{x}^2 + \bar{y}^1 + \bar{\bar{c}^0}^2)^2} + \overline{(x + \bar{x}^1 + \bar{y}^2 + \bar{\bar{c}^0}^2)^2}\Big]$$

It has to be noted here that the 2's of the function have not been taken as "don't cares" when obtaining F_1. The number of inputs for the carry out function can be slightly reduced if "don't cares" have been taken into consideration.

5. TERNARY STORAGE ELEMENTS AND SHIFT REGISTERS

Different types of ternary flip-flops (tri-flops): the PZN type, the clocked PZN, the D-type and the T-type tri-flops have been previously presented in [2].

Briefly, the PZN tri-flop may be constructed by cross-coupling two TNAND gates as shown in Fig. 7. The first TNAND has three inputs: P for setting the tri-flop to state "2" (positive), Z to zeroise its potential (state "1") and the third input coming from the output ($\bar{X}^1$) of the other TNAND to insure the regenerative property. The second TNAND gate has also three inputs: N for setting the tri-flop to state "0" (negative), Z to zeroise its potential and the third coming from the output X.

To set the tri-flop to state 2 we put a high to low pulse on the P input. A high to intermediate pulse on the Z input will set the device to the state 1 and to set it to the state 0 a high to low pulse is needed on the N input.

The PZN tri-flop can be controlled by clock signals through three TNAND gates one for each input. Each TNAND will have two inputs; the first for the P, Z or N terminal and the second for the clock C. The clock pulses and signals to P, Z and N inputs of the clocked PZN tri-flop are shown in Fig. 8.

Static ternary shift registers can be constructed using the clocked PZN tri-flop mentioned above or any of the D-type tri-flopspreviously described in [2].

The block diagram of a ternary arithmetic unit can be formed as in the binary case, by means of an adder and associated shift registers and gates.

6. CONCLUSION

A complete integrable ternary arithmetic unit can be designed using COS/MOS integrated circuits. The use of integrated circuits in the design of ternary digital machines

may offer the advantages of cost and wiring complexity reduction.

ACKNOWLEDGMENT

The authors are pleased to acknowledge the financial support of the National Research Council of Canada.

APPENDIX

Proof of De Morgan's theorems :

1- $\overline{(x+y)^0} = \overline{x^0} \cdot \overline{y^0}$

a) If $x = y = 0$

then $\overline{(x+y)^0} = \overline{(0+0)^0} = \overline{(0)^0} = 2$

and $\overline{x^0} \cdot \overline{y^0} = 2 \cdot 2 = 2$

therefore $\overline{(x+y)^0} = \overline{x^0} \cdot \overline{y^0}$

b) Let $x \neq 0$ and $y = 0$

then $\overline{(x+y)^0} = \overline{(x+0)^0} = \overline{x^0}$

but from definition $\overline{x^0} = 0$

and $\overline{x^0} \cdot \overline{y^0} = 0 \cdot 2 = 0$

therefore $\overline{(x+y)^0} = \overline{x^0} \cdot \overline{y^0}$

c) Let $x, y \neq 0$

then from definition $\overline{(x+y)^0} = 0$

and $\overline{x^0} \cdot \overline{y^0} = 0 \cdot 0 = 0$

therefore $\overline{(x+y)^0} = \overline{x^0} \cdot \overline{y^0}$

Q.E.D.

Similarly it can be proved that $\overline{(x \cdot y)^0} = \overline{x^0} + \overline{y^0}$

2- $\overline{(x+y)^1} = \overline{x^1} \cdot \overline{y^1}$

$$\overline{x^1} \cdot \overline{y^1} = \text{MIN}(\overline{x^1}, \overline{y^1})$$
$$= \text{MIN}(2-x, 2-y)$$
$$= 2 - \text{MAX}(x,y)$$
$$= 2 - (x+y)$$
$$= \overline{(x+y)^1}$$

Q.E.D.

Similarly it can be proved that $\overline{(x \cdot y)^1} = \overline{x^1} + \overline{y^1}$

3- $\overline{(x+y)^2} = \overline{x^2} \cdot \overline{y^2}$

a) If $x = y = 2$

then $\overline{(x+y)^2} = \overline{(2+2)^2} = \overline{(2)^2} = 0$

and $\overline{x^2} \cdot \overline{y^2} = 0 \cdot 0 = 0$

therefore $\overline{(x+y)^2} = \overline{x^2} \cdot \overline{y^2}$

b) Let $x \neq 2$ and $y = 2$

then $\overline{(x+y)^2} = \overline{(x+2)^2} = \overline{(2)^2} = 0$

and $\overline{x^2} \cdot \overline{y^2} = 2 \cdot 0 = 0$

therefore $\overline{(x+y)^2} = \overline{x^2} \cdot \overline{y^2}$

c) Let $x, y \neq 2$

then from definition $\overline{(x+y)^2} = 2$

and $\overline{x^2} \cdot \overline{y^2} = 2 \cdot 2 = 2$

therefore $\overline{(x+y)^2} = \overline{x^2} \cdot \overline{y^2}$

Q.E.D.

Similarly it can be proved that $\overline{(x \cdot y)^2} = \overline{x^2} + \overline{y^2}$

REFERENCES

[1] H.T. Mouftah and I.B. Jordan, "Integrated circuits for ternary logic", Proceeding of the 1974 International Symposium on Multiple-Valued Logic, May 1974, pp. 285-302.

[2] H.T. Mouftah and I.B. Jordan, "Design of ternary COS/MOS memory and sequential circuits", submitted for publication to the IEEE Trans. on Computers.

[3] H.T. Mouftah and I.B. Jordan, "Implementation of three-valued logic with COS/MOS integrated circuits", Electronics Letters, vol. 10, no. 21, October 1974, pp. 441-442.

[4] I. Halpern and M. Yoeli, "Ternary arithmetic unit", Proc. IEE, vol. 115, no. 10, October 1968, pp. 1385-1388.

[5] M. Yoeli and G. Rosenfeld, "Logical design of ternary switching circuits", IEEE Trans. Elect. Comp., vol. EC-14, February 1965, pp. 19-29.

[6] M. Bitran and M.J.O. Strutt, "Minimization of ternary logic and complete set of integrable circuits", Electronics and Communication, AEÜ, Band 25, 1971, pp. 387-392.

[7] R.S. Nutter and R.E. Swartwout, "A ternary logic minimization technique", Conference

Reccrd of the 1971 Symposium on the Theory and Applications of Multiple-Valued Logic Design, May 1971, pp. 112-123.

[8] R.S. Nutter, "The algebraic simplification of a ternary full adder", Conference Record of the 1972 Symposium on the Theory and Applications of Multiple-Valued Logic Design, May 1972, pp. 75-82.

[9] D.I. Porat, "Three-valued digital systems", Proc. IEE, vol. 116, no. 6, June 1969, pp. 947-954.

[10] S.Y. Su and A.A. Sarris, "The relationship between multivalued switching algebra and boolean algebra under different definitions of complement", IEEE Trans. on Computers, vol. C-21, no. 5, May 1972, pp. 479-485.

[11] COS/MOS Integrated Circuits Manual, RCA Technical Series CMS-271.

x	$\overline{x^0}$	$\overline{x^1}$	$\overline{x^2}$
2	0	0	0
1	0	1	2
0	2	2	2

Table I Truth table of ternary inverters

x	2	2	2	1	1	1	0	0	0
y	2	1	0	2	1	0	2	1	0
x + y	2	2	2	2	1	1	2	1	0
x · y	2	1	0	1	1	0	0	0	0

Table II Truth table of TOR and TAND operators

x	2	2	2	1	1	1	0	0	0
y	2	1	0	2	1	0	2	1	0
F(x,y)	2	2	0	0	1	0	1	1	2

(a) Truth table

	x = 2	x = 1	x = 0
y = 2	2	0	1
y = 1	2	1	1
y = 0	0	0	2

(b) Map

Table III Representations of a ternary function

x	y	x+y	$\overline{(x+y)^0}$	$\overline{(x+y)^1}$	$\overline{(x+y)^2}$	x · y	$\overline{(x \cdot y)^0}$	$\overline{(x \cdot y)^1}$	$\overline{(x \cdot y)^2}$
2	2	2	0	0	0	2	0	0	0
2	1	2	0	0	0	1	0	1	2
2	0	2	0	0	0	0	2	2	2
1	2	2	0	0	0	1	0	1	2
1	1	1	0	1	2	1	0	1	2
1	0	1	0	1	2	0	2	2	2
0	2	2	0	0	0	0	2	2	2
0	1	1	0	1	2	0	2	2	2
0	0	0	2	2	2	0	2	2	2

Table IV Truth table of TNOR and TNAND gates

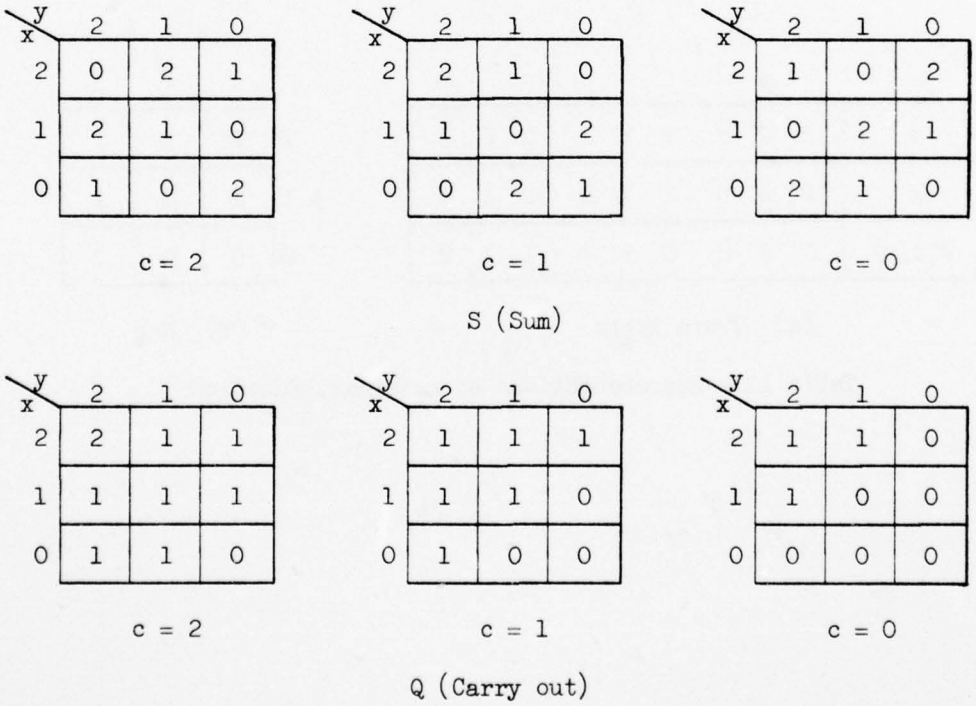
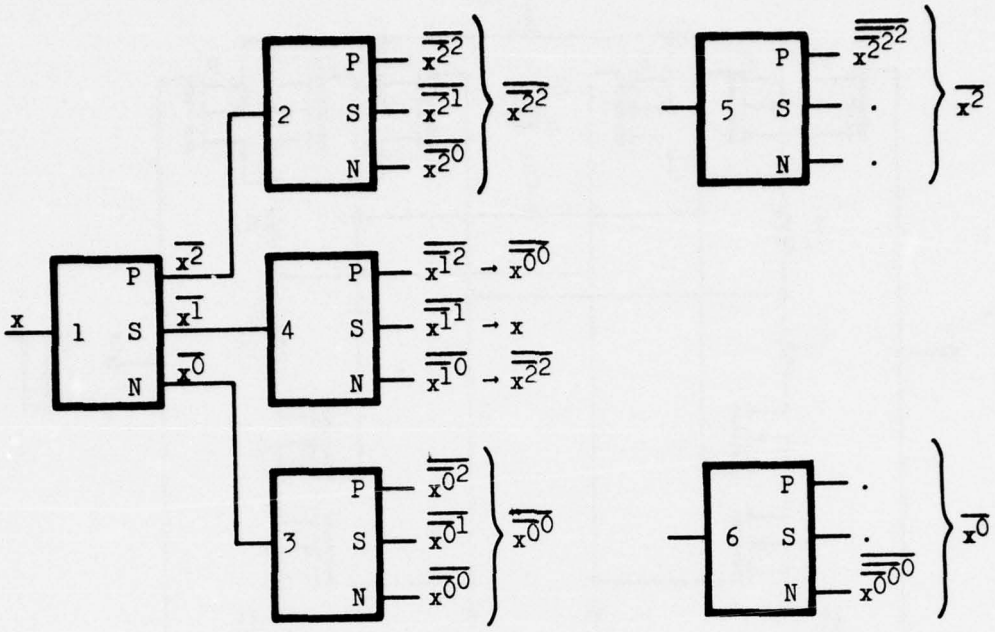
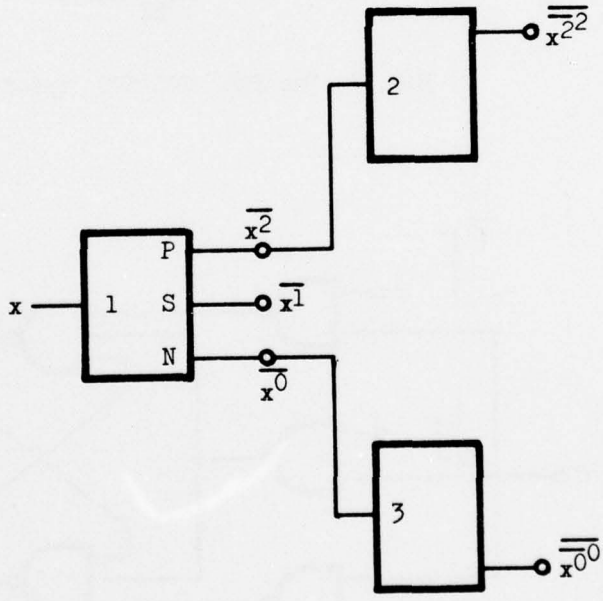


Fig. 6 Sum and Carry out of a stage of a ternary full adder



(a)



(b)

Fig. 5 a) Ternary inverters in cascade
b) The allowed connection of ternary inverters

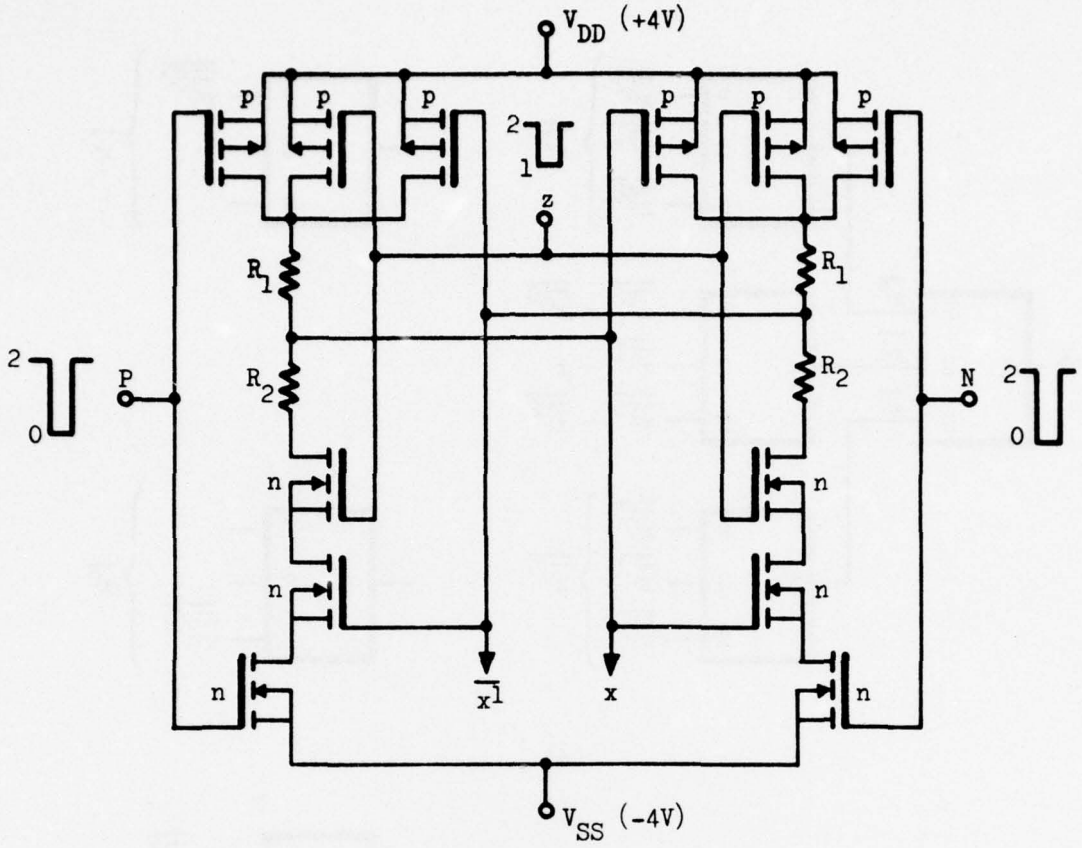


Fig. 7 The PZN COS/MOS tri-flop

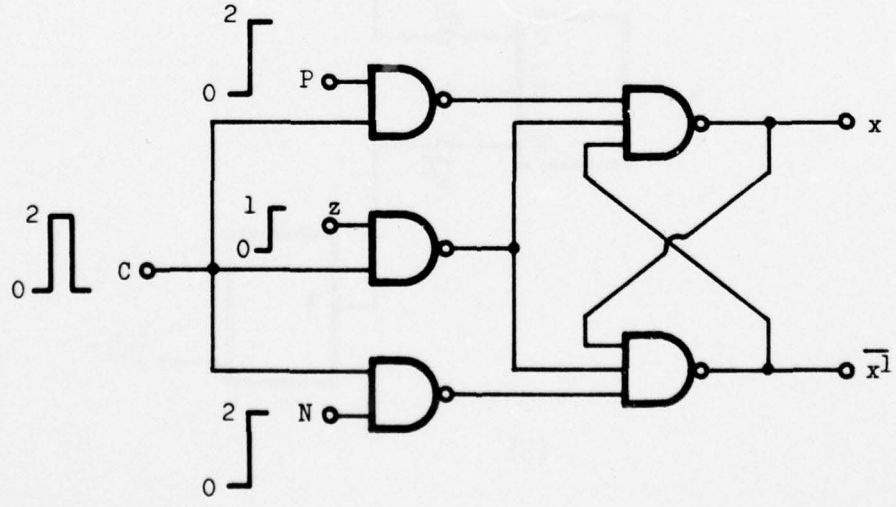


Fig. 8 The clocked PZN tri-flop

THRESHOLD LOGIC IN FAST TERNARY MULTIPLIERS*

Z.G. Vranesic and V.C. Hamacher

Departments of Electrical Engineering and Computer Science
University of Toronto
Toronto, Ontario

1. Introduction

In complex digital systems where the number of logic gates or packages and the number of interconnection wires is huge, it is reasonable to consider the use of multi-valued logic. The usual reasoning goes as follows: If more than two values can be represented on a single wire, the signals on such wires can potentially carry more "information", and the total wire count should be smaller than in an equivalent binary design. Further, if "more powerful" logic operations are available in the multi-valued logic environment, a given processing task should require fewer gates or packages. Therefore, total component and interconnection wire counts should be reducible by significant amounts.

The above introductory qualitative argument needs to be examined critically from a number of independent directions in order to establish any form of quantitative validity. One important aspect of the viability of multi-valued digital systems is the feasibility of circuit implementation of the supposed powerful primitives. This paper makes use of the recently developed many-valued multi-threshold {MT(R)} elements, which have been successfully tested in prototype form [1], [2].

We are concerned with an equally important aspect of the applicability of multi-valued logic in large digital systems. Namely, a synthesis technique is developed in a particular application area of contemporary interest that provides numerical comparisons between binary and multi-valued designs. This kind of evidence is a necessary component in any serious attempt at establishing the claim in the first paragraph. The specific application studied is combinational digital multipliers. A previous study, [3], examined this topic by assuming a complete set of multi-valued primitives of the class studied in [4], [5], [6], [7]. These logics can generally be viewed as extensions of the binary AND, OR, NOT system to the multi-valued environment; and consequently reference [3] presented comparisons on that basis. In this paper, the above mentioned MT(R) elements form the basis for the design. Since the MT(R) element is a natural generalization of the binary threshold element, comparisons to the binary situation have been based on the use of binary threshold elements wherever possible in the binary design.

A ternary (R = 3) multiplier design is developed in Section 2. The number representation chosen is balanced ternary, which uses base 3 positional notation and digits -1, 0, and +1. This choice will be seen to have certain advantages over the more conventional "0, 1, 2" positional notation. As well as having particular advantages in the present case, the use of balanced ternary adds to the area of investigations based on such an assumption [8], [9], [10].

---

* This research was supported in part by Canadian National Research Grants A5280 and A5192.

2. A Balanced Ternary Multiplier Using MT(3) Elements

It is helpful to deal with an example in order to illustrate the algorithm that is proposed for implementation of combinational balanced ternary multipliers. Figure 1(a) shows the usual "paper-and-pencil" algorithm for the multiplication of two particular 8-digit balanced ternary numbers, hereafter called simply ternary numbers. The conventional sign and magnitude base 10 representation of the value of a ternary number is (using the multiplicand from Figure 1(a)):

$$-1001\text{-}1\text{-}101$$

$$= (-1)\times 3^7+(0)\times 3^6+(0)\times 3^5+(1)\times 3^4+(-1)\times 3^3+(-1)\times 3^2+(0)\times 3^1+(1)\times 3^0$$

$$= -2187 + 0 + 0 + 81 - 27 - 9 + 0 + 1$$

$$= -2141$$

A basic technique for speeding up multiplication is the parallel carry-save reduction of all summands. This technique, first presented by Wallace [11] for the binary case, generalizes to the ternary case in the form of 4 to 2 reduction of summands. It is clear that the sum of four ternary digits, all weighted 3^k for some $k \geq 0$, can be represented by the two ternary digits s (sum), weighted 3^k, and c (carry), weighted 3^{k+1}. Iterating this scheme until only two summands are left then allows the final product to be developed by a 2n digit ternary adder in the case of n digit operands. Figure 1(b) shows how this algorithm operates on the operands of Figure 1(a). The generalization to other operand lengths is straight forward. A block diagram of the logic components needed for an operand length of n = 16 is shown in Figure 2. The new MT(3) elements are used extensively in this design in addition to the use of ternary MIN and MAX gates for bussing purposes. Summand selections based on multiplier digit values is carried out at the top of the figure. A simple one-input MT(3) gate is used to develop $\bar{d}$. In general, an MT(R) element implements the R-valued switching function of n variables as follows [1], [2]: $f(x_1, x_2, \ldots, x_n) = h(e)$ where $e = \sum_{i=1}^{n} a_i x_i \equiv$ excitation, $a_i \equiv$ integer weight associated with input x_i, $h(e) \equiv$ transfer characteristic of the element (i.e., the output value when the excitation is e), with f, h, $x_i \in \{0, 1, \ldots, (R-1)\}$. The transfer characteristic h(e) may be described graphically as in Figure 3, where a 3-valued function of 2 variables is presented with $a_1 = 2$ and $a_2 = 1$. Figure 3(a) gives the truth table and 3(b) and 3(c) show the h(e) function and element schematic, respectively. The notation inside of the element schematic is a code for h(e) values as e ranges from its lowest to highest values. The code says: h = 1 for the lowest e value, h = 2 for the next e value, 0 for the next, 1 for the next 3, and finally 2 for the last e value.

In the middle of Figure 2, a cascade of 4→2 reducers is shown that achieves the reduction of the summands to two vectors as described previously. Figure 4 shows the functional specification of a 4 to 2 reducer, where the desired sum and carry outputs are defined in terms of the input excitation. It should be noted that this corresponds to a general 4-input ternary full-adder, where the carry digit may take on all three logic values.

While it is apparent that two MT(3) gates can be used to implement the s and c functions directly, such an approach is not attractive, since the gate needed to realize s requires eight thresholds, which adversely affects both the cost and propagation delay. A better arrangement is given in Figure 5. This circuit requires

MT(3) gates with two thresholds each, making use of the fact that the s and c outputs have essentially the same patterns, where the s output has three times the frequency of the c output. We should emphasize that this general notion is highly useful in most arithmetic applications.

The last subsystem to be considered in the general schematic of Figure 2 is the adder that generates the product from the final 2 operand vectors. It is possible to specify a ripple-carry adder that operates directly on the -1, 0, and +1 values of these operands, but it would seriously degrade the speed of the whole system. It is also possible to implement a variation of the usual carry - lookahead technique for speeding up adders, but it is complicated because the carries can have any of the values -1, 0, or +1, not simply 0 or 1 as in the case of {0, 1, ... , R-1} -digit positional adders for two operands. The complications lead to relatively complex logic arrangements, and it is not obvious what the practical speed gains can be. We will describe and use another possibility that is a three-step process, First, the two (-1, 0, +1) operands are converted to a (0, 1, 2) representation; next, they are added in a lookahead adder to generate a 0, 1, 2 representation that finally can be easily converted to the correct (-1, 0, +1) form of the product. Each of the processes is fast enough so that there is a big improvement over the direct ripple-carry (-1, 0, +1) process, and the total scheme should be cost and delay competitive with the (-1, 0, +1) carry-lookahead possibility.

In more detail, the workings of this addition technique are best explained by following an example. Figure 6(a) shows the addition of a pair of (-1, 0, +1) vectors. The conversion of these vectors to a (0, 1, 2) form is shown in Figure 6(b). The conversion is the addition of the value 1 to <u>each</u> digit position of <u>each</u> of the two vectors, along with the addition of a third vector of 1's to the converted operands. The addition of 1's to the digits of the operands simply shifts the (-1, 0, +1) value range to the (0, 1, 2) value range and does not generate any carries. In fact this merely amounts to a re-interpretation of the electrical signal values defining the two operands. Considering that these operands are now (0, 1, 2) vectors, the <u>actual</u> addition of the third vector of 1's is done in a 3→2 reducer whose design is very similar (and simpler) to the 4→2 reducers in Figures 4 and 5. The low order digit position is actually a 4→2 reducer to accomodate an extra 1 in the low-order position. At this point it is necessary to state how we have changed the value of the product being computed. The "conceptual" addition of 1's to the two operand vectors and the low-order extra 1 amount to numerically adding 1 in position k in Figure 6(a). Thus, if we actually add 1's in the 3→2 reduction process, then add the two operand vectors using a conventional (0, 1, 2) lookahead addition process, and later (in the third phase of the scheme) subtract 1's (by conceptually shifting the answer from (0, 1, 2) to (-1, 0, +1) value ranges) we will be left with the correct product in (-1, 0, +1) representation in positions k-1 through 0 and we can ignore the 1 that was produced in position k. The conversion, 3→2 reduction and (0, 1, 2) addition is shown in Figure 6(b), with final subtraction of 1's being done in part (c) to get the desired product. The actual addition process can be done with conventional lookahead techniques. It has been shown [1] that digit grouping in pairs before specifying Propagate and Generate functions is a speed and cost effective design that utilizes simple MT(3) gates.

3. Comparison with Binary Multipliers

Binary threshold-type gates are the natural primitive blocks to be used in constructing binary multipliers for comparison with the proposed ternary designs. The class of basic block chosen here is one that has recently been described by both Swartzlander [12] and Hampel [13]. They give electronic realizations of such circuits. In [12], a binary 7→3 reducer is presented. This is a straightforward extension of the Wallace [11] 3→2 reduction process. Seven unity-weight input lines are converted into 3 output lines weighted 4, 2, and 1 that indicate a count of the number of logic ones on the seven input lines. The use of such blocks in the carry-save reduction tree process is obvious. Figure 7 shows the schematic of a 24-bit binary multiplier that utilizes 7→3 reducers as well as 6→3 reducers. The thirteen summands result from bit-pair grouping of the multiplier using the Wallace [11] recoding technique. This technique, along with the final adder process, has been used and explained in [3].

In addition to using the threshold circuits as 7→3 reducers, we have also made use of threshold gates to obtain the Propagate and Generate functions in the carry-lookahead adder. This was done in the same manner as previously pointed out for the ternary case, namely by digit grouping the summands in pairs. Thus, the number of required Propagate and Generate functions is halved, resulting in a much simpler AND-OR network for final derivation of carry signals.

When comparing the ternary and binary multipliers that have been presented, the natural criteria are speed and cost. It is not easy to get general agreement on realistic values for these parameters when non-trivial unconventional building blocks like MT(R) elements and 7→3 reducers are used. Nevertheless, we propose the following.

Cost: Both "gate" and "input" costs are computed. The unit of gate cost is that of a binary AND, OR, or NOT gate or a ternary MAX, MIN, or INVERTER gate. The unit of input cost is an input wire to a primitive block whether it is a binary OR input, or a ternary 4→2 reducer input, or a binary 7→3 reducer input, etc. Cost estimates for the larger binary and ternary blocks are given in Table I. The k+1 value for MT(3) blocks with k thresholds, and the n+m value for n→m reducers are derived from considerations of the structure of the electronic implementations shown in [2] and [12], respectively. In particular, each threshold requires a comparator, and there are n of these needed in an n→m reducer. In addition, an n→m reducer requires conventional binary logic operating on the comparator outputs to generate the m block outputs, resulting in the "m" term. An MT(3) block is assigned a cost of k+1 for k thresholds plus some output circuitry.

Delay: The unit of delay is that of a binary AND, OR, or NOT gate or a ternary MAX, MIN, or INVERTER gate. We define this to be τ seconds. In the case of n→m reducers and MT(3) gates it is possible to assume that their delay is proportional to the number of thresholds, with the extra 1 term in the n→m reducer block because of the output logic delay.

Using these assumptions a table of overall system cost and delay can be constructed for comparable length binary and ternary multipliers. We have chosen binary multipliers with operand lengths of 12, 24, and 32 bits. The corresponding ternary multipliers have operand lengths of 8, 16, and 21 digits, respectively. This results from the assumption that if T ternary digits are to represent the same number of values representable by b binary bits, then $3^T \geq 2^b$. This solves to $T = \lceil 0.631\ b \rceil$ and generates the above values.

Details of subtotals for the 24-bit vs. 16-digit ternary comparison are given in Table II and Table III shows the totals for the range of operands stated above.

4. Conclusions

Design examples of this paper indicate the attractiveness of threshold functions in the implementation of parallel multipliers, particularly as part of the carry-save reduction tree and the final carry-lookahead adder.

Furthermore, it is apparent that a balanced ternary multiplier can be constructed to handle operands of equivalent length in approximately the same time delay, but which requires 35% fewer (binary equivalent) gates and 40% fewer inputs than the corresponding binary scheme. This points out the benefits that can be derived from the utilization of balanced ternary coding techniques and multi-threshold circuits.

We should note that the comparison between binary and ternary threshold implementations is quite natural. The necessary basic circuits tend to be constructed along similar lines, usually employing some form of current switching by means of differential pairs. Thus, the problems in constructing ternary circuits are not much different from those encountered with the binary ones.

Finally, it is worthwhile reemphasizing the fact that threshold gates can reduce the complexity of carry-lookahead circuits, regardless of the radix used. This should not be surprising since carry functions, and indeed most arithmetic functions, exhibit threshold characteristics.

References

[1] A. Druzeta, "Many-valued Multi-threshold Logic", Ph.D. Thesis, Department of Electrical Engineering, University of Toronto, November, 1974.

[2] A. Druzeta, Z.G. Vranesic, and A.S. Sedra, "Application of Multi-threshold Elements in the Realization of Many-valued Logic Networks", IEEE Trans. Computers, Vol. C-23, November 1974, pp. 1194-1198.

[3] Z.G. Vranesic and V.C. Hamacher, "Ternary Logic in Parallel Multipliers", The Computer Journal, Vol. 15, No. 3, November 1972, pp. 254-258.

[4] Z.G. Vranesic, E.S. Lee, and K.C. Smith, "A Many-valued Algebra for Switching Systems", IEEE Trans. Computers, Vol. C-19, October 1970, pp. 964-971.

[5] E.L. Post, "Introduction to a General Theory of Elementary Propositions", Amer. J. Math., Vol. 43, 1921, pp. 163-185.

[6] S.V. Yablonskij, "Functional Constructions in K-valued Logic", Proc. of Steklov Inst. of Math., (in Russian), Vol. 51, 1958, pp. 5-142.

[7] Yu. L. Ivaskiv, "Principles of Multi-valued Implementation Schemes", (book in Russian), Naukova Dumka, Kiev 1971.

[8] G. Frieder, A. Fong, and C.Y. Chao, "A Balanced Ternary Computer", Proc. 1973 International Symposium on Multiple-valued Logic, Toronto, Ontario, May 1973, pp. 68-88.

[9] I. Halpern and M. Yoeli, "Ternary Arithmetic Unit", Proc. IEE, Vol. 115, No. 10, October 1968, pp. 1385-1388.

[10] P. Sebastian and Z.G. Vranesic, "Ternary Logic in Arighmetic Units", Proc. 1972 Symposium on the Theory and Applications of Multiple-valued Logic Design, Buffalo, May 1972, pp. 153-159.

[11] C.S. Wallace, "A Suggestion for a Fast Multiplier", IEEE Trans. Elect. Comp., Vol. EC-13, 1964, pp. 14-17.

[12] E.E. Swartzlander, Jr., "The Quasi-Serial Multiplier", IEEE Trans. on Computer Vol. C-22, No. 4, April 1973, pp. 317-321.

[13] D. Hampel, "Multifunction Threshold Gates", IEEE Trans. on Computers, Vol. C-22, No. 2, February 1973, pp. 197-203.

AD-A045 757 INDIANA UNIV BLOOMINGTON DEPT OF COMPUTER SCIENCE F/G 9/2
PROCEEDINGS OF THE 1975 INTERNATIONAL SYMPOSIUM ON MULTIPLE-VAL--ETC(U)
MAY 75 G EPSTEIN N00014-75-C-0449
UNCLASSIFIED MVL-75-001 NL

5 OF 6
AD A045757



TABLE I

Logic Cost and Delay Values.

Block	Gate Cost	Delay
binary AND, OR, or NOT	1	τ
ternary MAX, MIN, INVERTER	1	τ
ternary MT(3) with k thresholds	k+1	$k\tau$
binary n→m reducers	n+m	$(n+1)\tau$

TABLE III

Cost and Delay Comparisons Among
Binary and Ternary Multipliers

Operand Lengths	Unit Gate Costs		Inputs		Delay (τ)	
	Binary	Ternary	Binary	Ternary	Binary	Ternary
Binary = 12 Ternary = 8	946	615	1,915	1,088	25	27
Binary = 24 Ternary = 16	3,151	2,035	6,228	3,542	34	33
Binary = 32 Ternary = 21	5,114	3,420	9,992	5,866	41	37

TABLE II

Multiplier Cost and Delay Values by Subsystem in the 24-bit Binary vs. 16-digit Ternary comparison.

Subsystem	24-bit binary multiplier							16-digit ternary multiplier						
	AND OR NOT gates	m→n reducers		threshold gates	UNIT GATE COST	INPUTS	DELAY (τ)	MAX MIN INV. gates	$MT_k(3)$* gates		UNIT GATE COST	INPUTS	DELAY (τ)
		m→n	#						k	#			
Multiplier recoding and summand selection	1,564	-	-	-	1,564	3,745	4	768	2	16	816	1,552	4
Tree reduction	-	7→3 6→3 5→3 4→3 3→2	12 32 12 12 57	-	120 288 96 84 285	84 192 60 48 171	19	-	2	312	936	1,268	16
Final Adder	618	-	-	48	714	1,928	11	75	2 1	48 32	144 64 75	288 128 306	13
TOTALS	2,182		125	48	3,151	6,228	34	843		408	2,035	3,542	33

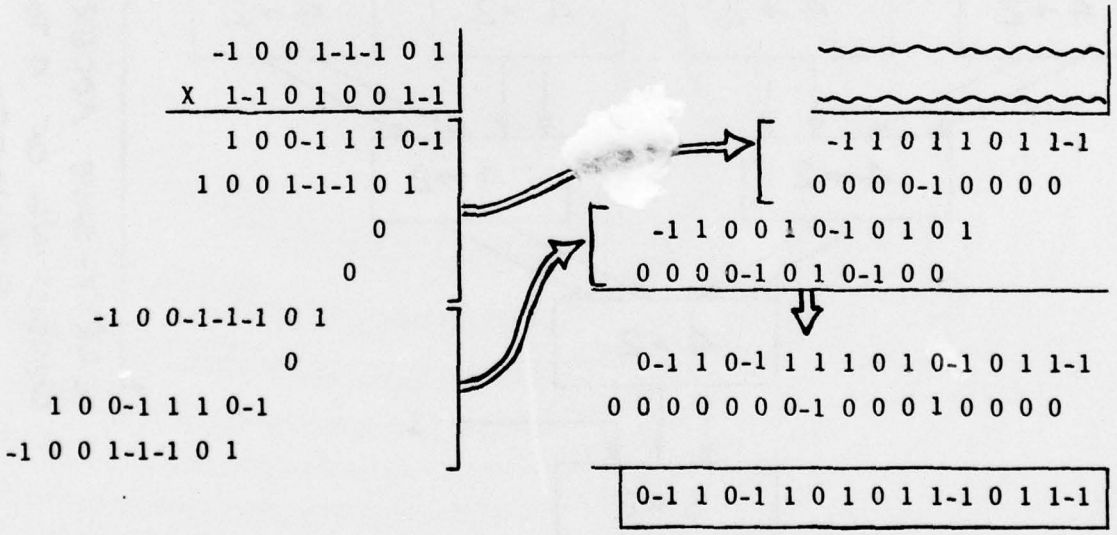
* $MT_k(3)$ is an MT(3) gate with k thresholds.

```
                                  -1  0  0  1 -1 -1  0  1
                               X   1 -1  0  1  0  0  1 -1
                                  ---------------------------
                                   1  0  0 -1  1  1  0 -1
                                -1  0  0  1 -1 -1  0  1
                       -1  0  0  1 -1 -1  0  1  0  0
                  1  0  0 -1  1  1  0 -1  0
             -1  0  0  1 -1 -1  0  1
          0 -1  1  0 -1  1  0  1  0  1  1 -1  0  1  1 -1
```

(a) Usual shift and add algorithm

```
          -1 0 0 1-1-1 0 1
        X  1-1 0 1 0 0 1-1
          ---------------
           1 0 0-1 1 1 0-1
         1 0 0 1-1-1 0 1
                     0
                   0
     -1 0 0-1-1-1 0 1
                 0
   1 0 0-1 1 1 0-1
-1 0 0 1-1-1 0 1
```

```
-1 1 0 1 1 0 1 1-1
0 0 0 0-1 0 0 0 0
-1 1 0 0 1 0-1 0 1 0 1
0 0 0 0-1 0 1 0-1 0 0
0-1 1 0-1 1 1 1 0 1 0-1 0 1 1-1
0 0 0 0 0 0 0-1 0 0 0 1 0 0 0 0
0-1 1 0-1 1 0 1 0 1 1-1 0 1 1-1
```



(b) Parallel carry-save reduction algorithm

Figure 1: Balanced ternary multiplication examples

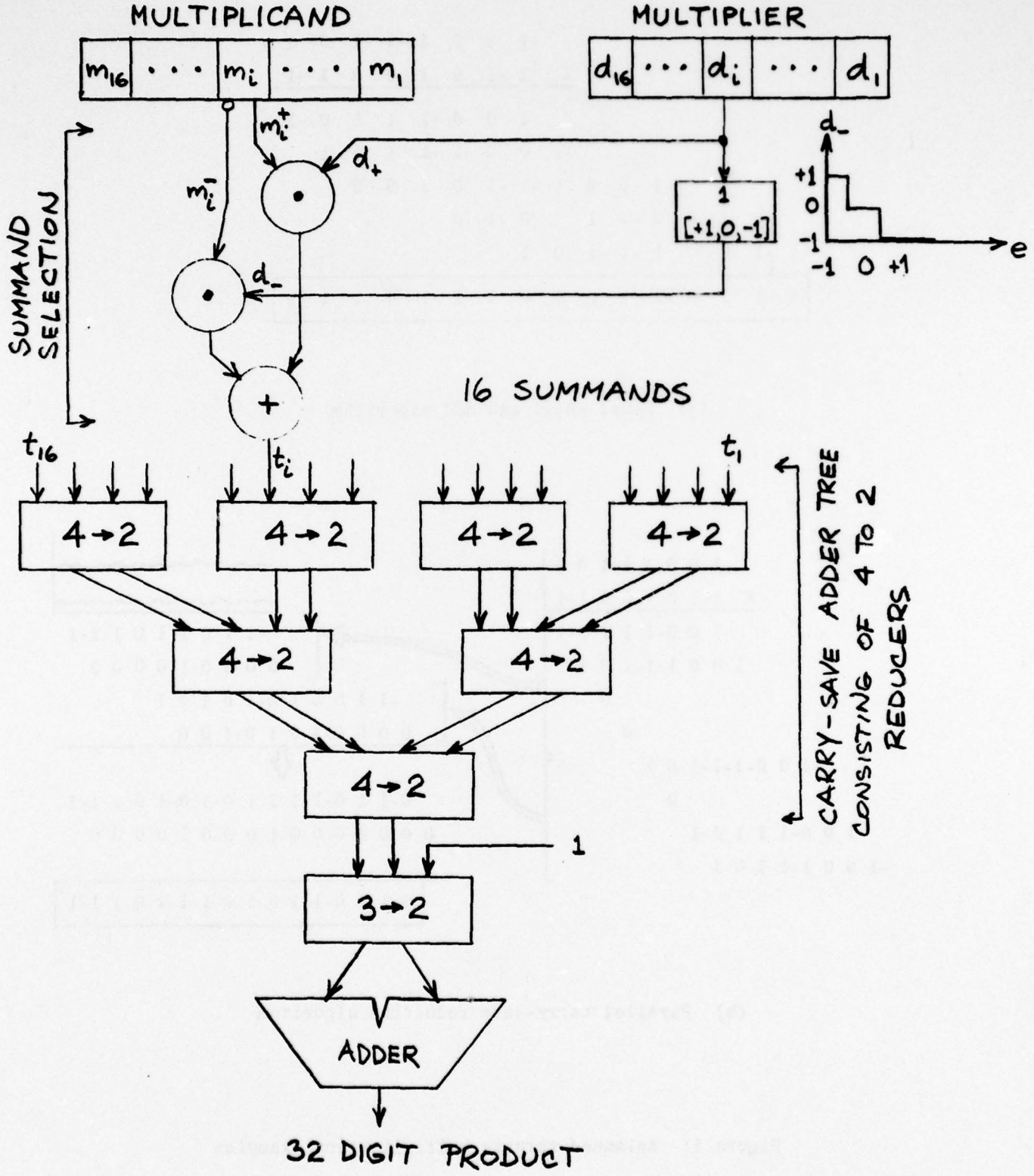
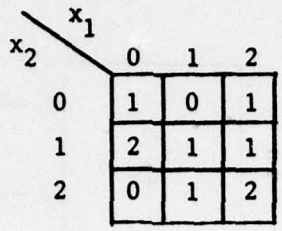
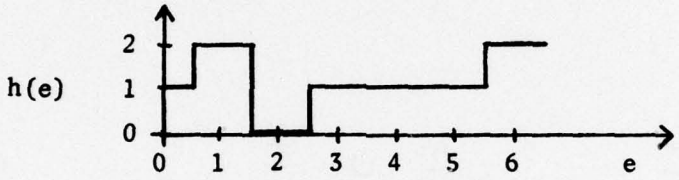


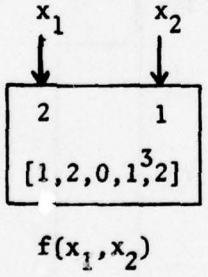
Figure 2: A Balanced Ternary 16-digit Multiplier



(a) Truth table to be implemented



(b) Transfer characteristic



(c) Element schematic

Figure 3: MT(3) element example

$e = \sum_{i=1}^{4} x_i$	-4	-3	-2	-1	0	+1	+2	+3	+4
S	-1	0	+1	-1	0	+1	-1	0	+1
C	-1	-1	-1	0	0	0	+1	+1	+1

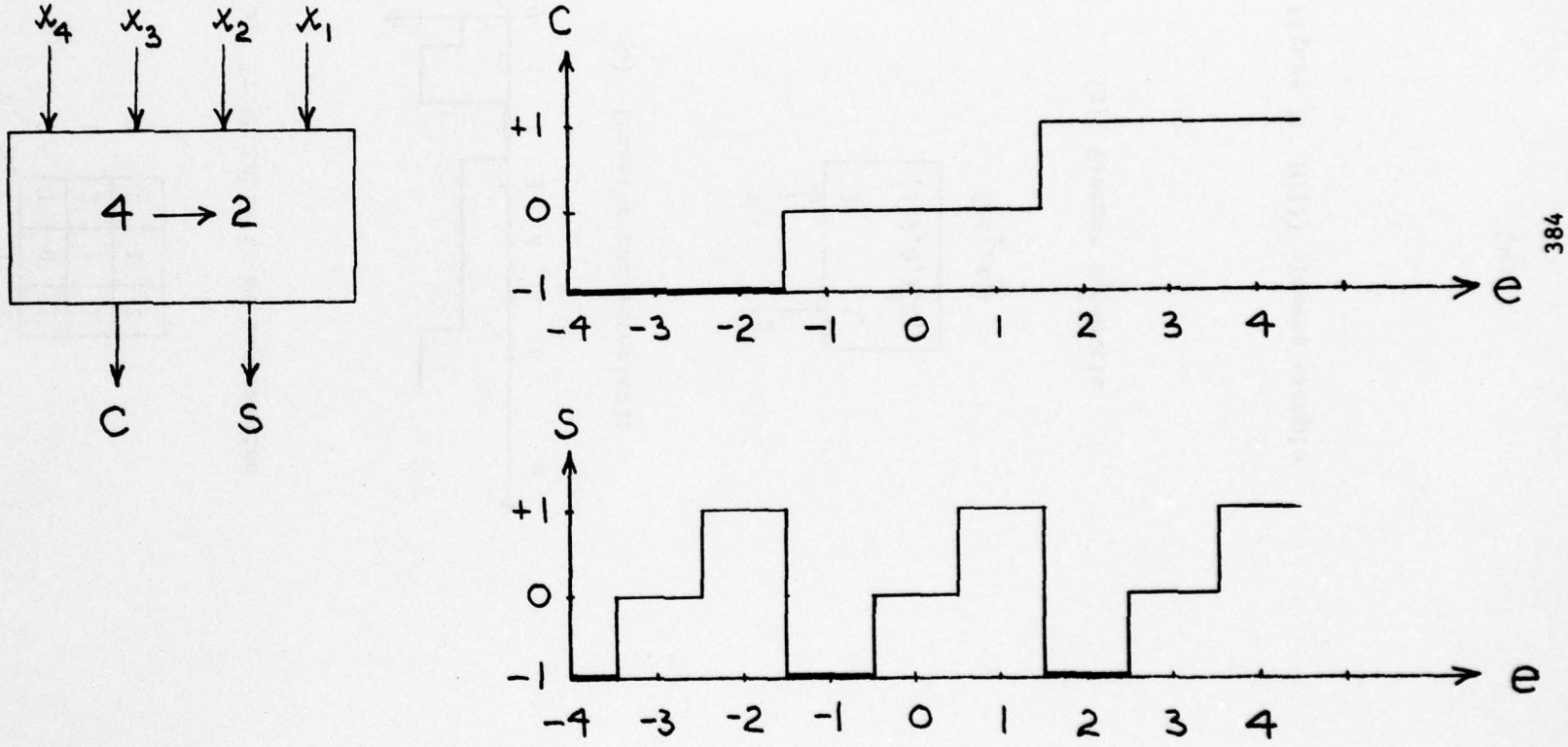
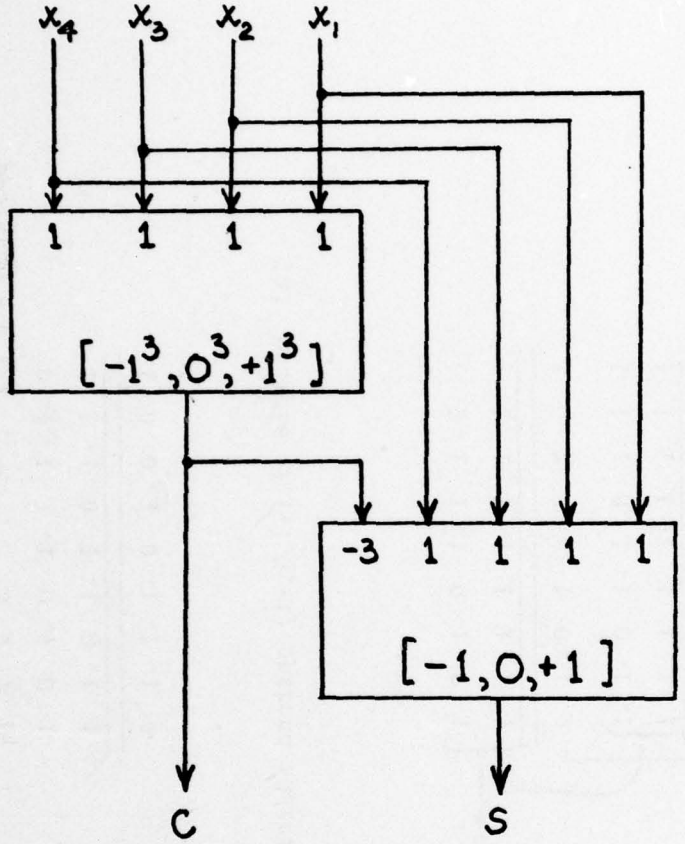


Figure 4. Functional Requirements of a 4 to 2 Reducer.



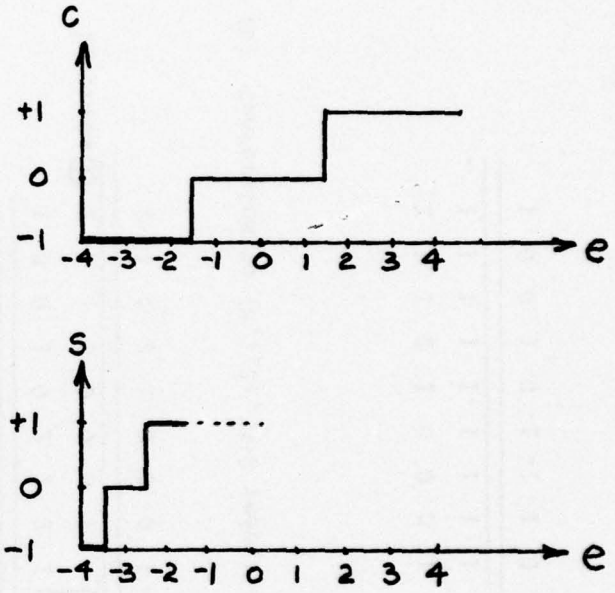
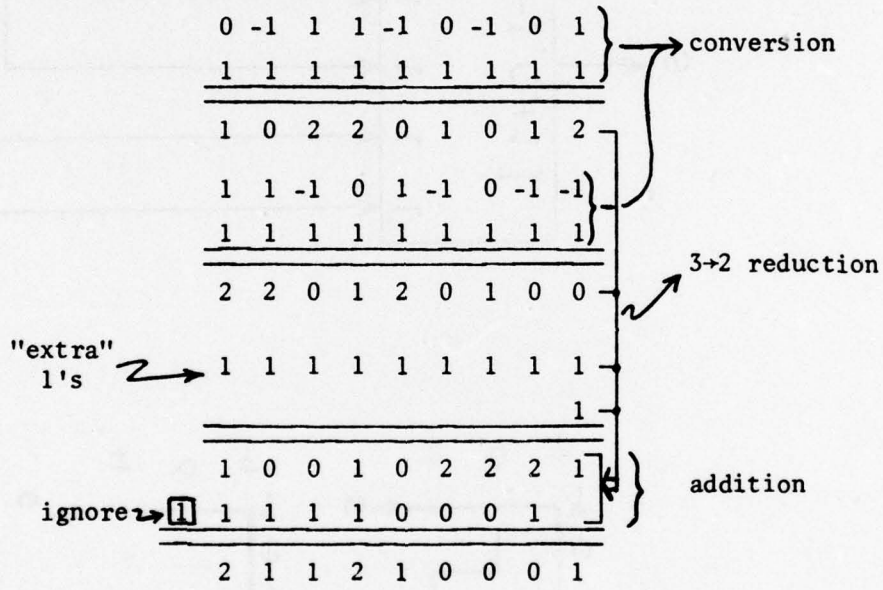


Figure 5. 4 to 2 Reducer Circuit

position =	k	k-1	.	.	.	3	2	1	0
	0	-1	1	1	-1	0	-1	0	1
	1	1	-1	0	1	-1	0	-1	-1
	1	0	0	1	0	-1	-1	-1	0

(a) Example of (-1,0,+1) operand addition



(b) Conversion to (0,1,2), 3→2 reduction, addition

	2	1	1	2	1	0	0	0	1
−	1	1	1	1	1	1	1	1	1
	1	0	0	1	0	-1	-1	-1	0

(c) Subtraction of 1's to yield (-1,0,+1) answer

Figure 6: Final adder process example

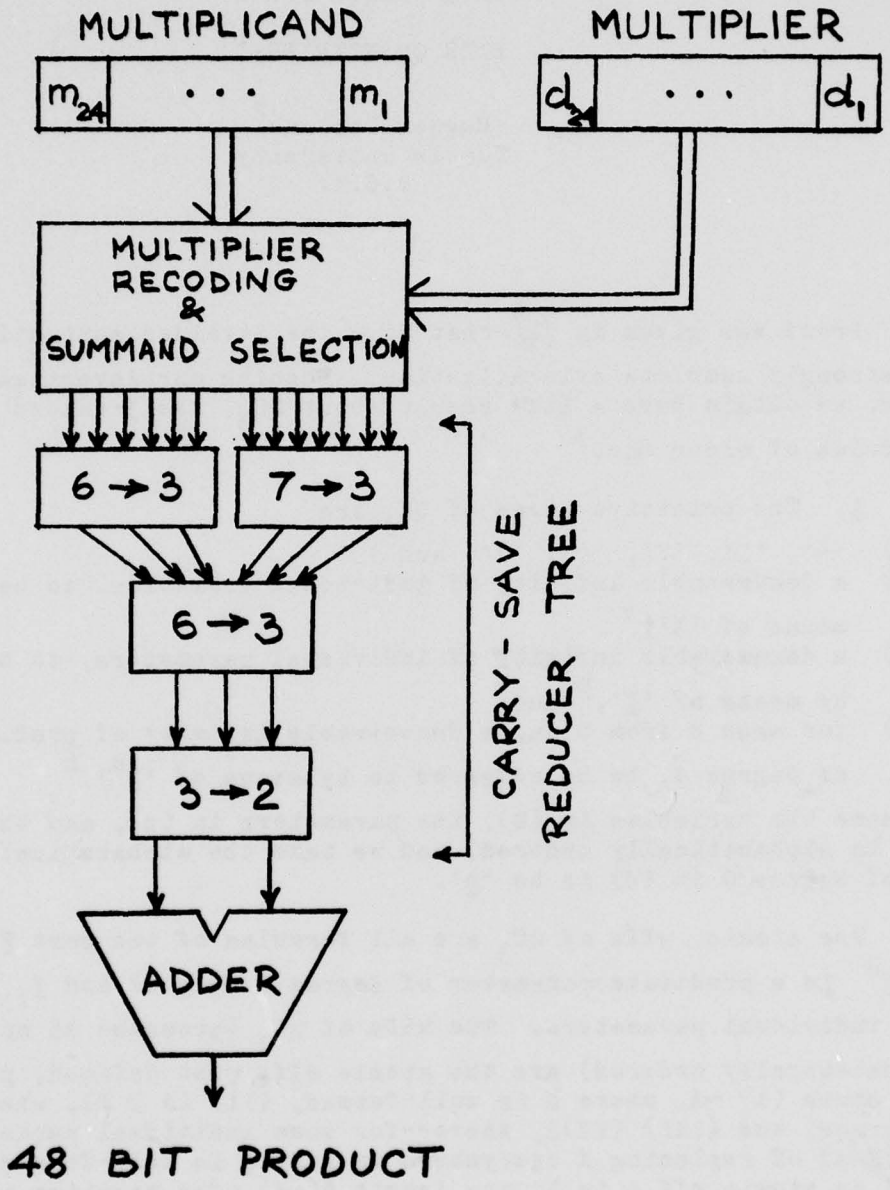


Figure 7: A Binary 24-bit Multiplier

A HENKIN-TYPE COMPLETENESS PROOF FOR 3-VALUED LOGIC WITH QUANTIFIERS[1]

Hugues Leblanc[2]
Temple University
U.S.A.

Proof was given in [1] that SC_3, the 3-valued sentential calculus, has a strongly complete axiomatization. Pushing our investigation one step further, we obtain here a like result about QC_3, the 3-valued quantificational calculus of order one.[3]

1. The primitive signs of QC_3 are

(a) '~', '⊃', '∀', '(', ')', and ',',
(b) a denumerable infinity of individual variables, to be referred to by means of 'X',[4]
(c) a denumerable infinity of individual parameters, to be referred to by means of '$\underline{X}$',[5] and
(d) for each d from 0 on, a denumerable infinity of predicate parameters of degree d, to be referred to by means of '$\underline{F}^d$'.[6]

We presume the variables in (b), the parameters in (c), and the parameters in (d) to be alphabetically ordered; and we take the alphabetically first parameter of degree 0 in (d) to be '$\underline{p}$'.

The atomic wffs of QC_3 are all formulas of the sort $\underline{F}^d(\underline{X}_1,\underline{X}_2,\ldots,\underline{X}_d)$, where $\underline{F}^d$ is a predicate parameter of degree d ($d \geq 0$) and $\underline{X}_1$, $\underline{X}_2$, ..., and $\underline{X}_d$ are individual parameters. The wffs of QC_3 (presumed at one point below to be alphabetically ordered) are the atomic wffs just defined, plus all formulas of the sorts (i) ~A, where A is well-formed, (ii) (A ⊃ B), where A and B are well-formed, and (iii) (∀X)A, where--for some individual parameter $\underline{X}$--the result $A(\underline{X}/X)$ of replacing X everywhere in A by $\underline{X}$ is well-formed.[7] The length $\ell(A)$ of an atomic wff A is 1; the length $\ell(\sim A)$ of a negation ~A is $\ell(A)+1$; the length $\ell((A \supset B))$ of a conditional (A ⊃ B) is $\ell(A)+\ell(B)+1$; and the length $\ell((\forall X)A)$ of a quantification (∀X)A is $\ell(A(\underline{X}/X))+1$, where $\underline{X}$ is the alphabetically earliest individual parameter of QC_3.

We avail ourselves of the following ten abbreviations:

$$'f' =_{df} '\sim(\underline{p} \supset \underline{p})'$$
$$(A \vee B) =_{df} ((A \supset B) \supset B)^8$$
$$(A \,\&\, B) =_{df} \sim(\sim A \vee \sim B)$$
$$(A \equiv B) =_{df} ((A \supset B) \,\&\, (B \supset A))$$

$$(A \; I \; B) =_{df} (A \supset (A \supset B))$$
$$-A =_{df} (A \supset \sim A)^{9}$$
$$J_1(A) =_{df} \sim(A \supset \sim A)$$
$$J_3(A) =_{df} \sim(\sim A \supset A)$$
$$J_2(A) =_{df} \sim(J_1(A) \vee J_3(A))$$
$$(\exists X)A =_{df} \sim(\forall X)\sim A;$$

and we omit outer parentheses whenever clarity permits.

Sets of wffs play a major role in the paper. We take an individual parameter to be foreign to a set S of wffs if the parameter does not occur in any member of the set, and we declare S infinitely extendible if $aleph_0$ individual parameters are foreign to S. Given a mapping M of one set of individual parameters into another, we understand by the M-rewrite of a wff A the result of simultaneously replacing in A all individual parameters from the first set by their respective values under M; and we understand by the M-rewrite of a set S of wffs $\emptyset$ when S is empty, otherwise the set consisting of the M-rewrites of the various members of S. Lastly, given two sets S and S′ of wffs, we declare S′ isomorphic to S if--for some one-to-one mapping M of the individual parameters of QC_3 occurring in S into all the individual parameters of QC_3--S′ is the M-rewrite of S.

The axioms of QC_3 are all wffs of the sorts A1-A4 on p. 325 of [1], plus all those of the sorts

A5. $(\forall X)(A \supset B) \supset ((\forall X)A \supset (\forall X)B)$,

A6. $A \supset (\forall X)A$,[10]

A7. $(\forall X)A \supset A(\underline{X}/X)$,

plus all those of the sort $(\forall X)A$, where--for some individual parameter $\underline{X}$ foreign to $(\forall X)A$--$A(\underline{X}/X)$ is an axiom of QC_3. The notions of provability, syntactic (in) consistency, and maximal consistency are then defined as on pp. 325-6 of [1], but with 'QC_3' substituting throughout for 'SC_3'.

Our truth-values are (the designated) 1 and (the undesignated) 2 and 3.[11] Truth-value assignments are functions from the atomic wffs of QC_3 to $\{1,2,3\}$, and the truth-values under these of negations and conditionals are reckoned as on p. 326 of [1].[12] As for quantifications, $(\forall X)A$ evaluates to 1 under a truth-value assignment α if $A(\underline{X}/X)$ does so for every individual parameter $\underline{X}$ of QC_3; $(\forall X)A$ evaluates to 3 under α if $A(\underline{X}/X)$ does so for at least one individual parameter $\underline{X}$ of QC_3; otherwise, $(\forall X)A$ evaluates to 2 under α.[13] We take a set S of wffs to be truth-value verifiable if there is a truth-value assignment under which all members of S evaluate to 1; we take S to be semantically consistent if either S or some set isomorphic to S is truth-value verifiable;[14] we take S to entail a wff A if $S \cup \{-A\}$ is semantically inconsistent; and we take the wff A to be valid if $\emptyset$ entails A.

2. Our completeness proof, as extension of that in [1], uses five fresh results: L3(c) and L4(a)-(d) below. Proof of L3(c) can be recovered from [1], pp. 336-337, and so is omitted here; but proofs of L4(a)-(d) are given in full.

Our first lemma is L1 in [1], pp. 326-7, which we shall presume the reader to have on hand. Our second lemma deals with truth-functional matters, and our third with quantificational ones.

L2. (a) If $S \vdash A \supset B$, then $S \vdash (B \supset C) \supset (A \supset C)$.

(b) If $S \vdash A \supset B$ and $S \vdash B \supset C$, then $S \vdash A \supset C$.

(c) If $S \vdash \sim A \supset \sim B$, then $S \vdash B \supset A$.

(d) If $S \vdash A \supset B$ and $S \vdash \sim B$, then $S \vdash \sim A$.

(e) If $S \cup \{A\} \vdash B$ and $S \vdash A' \supset A$, then $S \cup \{A'\} \vdash B$.

(f) If $S \vdash A \vee B$, then $S \vdash B \vee A$.

(g) If $S \vdash A \vee B$ and $S \vdash A \supset A'$, then $S \vdash A' \vee B$.

(h) If $S \vdash A \vee B$ and $S \vdash A \supset A'$, then $S \vdash (A' \,\&\, A) \vee B$.

(i) If $S \vdash A \vee B$ and $S \vdash B \supset B'$, then $S \vdash A \vee B'$.

(j) If $S \vdash A \vee (B \vee C)$ and $S \vdash B \supset B'$, then $S \vdash A \vee (B' \vee C)$.

(k) If $S \vdash A \vee (B \vee C)$ and $S \vdash C \supset C'$, then $S \vdash A \vee (B \vee C')$.

(ℓ) If $S \vdash A \vee (B \,\&\, C)$ and $S \vdash C \supset C'$, then $S \vdash A \vee (B \,\&\, C')$.

(m) If $S \cup \{C\} \vdash A \vee B$, then $S \cup \{C\} \vdash A \vee (B \,\&\, C)$.

(n) If $S \cup \{C\} \vdash A \vee (B \vee \sim C)$, then $S \cup \{C\} \vdash A \vee B$.

(o) If $S \vdash A \;\mathrm{I}\; -A$, then $S \vdash -A$.

(p) If $S \vdash J_1(A) \vee J_2(A)$, then $S \vdash \sim J_3(A)$.

(q) $S \vdash \sim J_3(A) \supset (J_1(A) \vee J_2(A))$.

(r) $S \vdash -J_3(A) \supset \sim J_3(A)$.

(s) If $S \cup \{J_3(A)\} \vdash B$, then $S \vdash J_3(A) \supset B$.

Proof: (a) Since $(A \supset B) \supset ((B \supset C) \supset (A \supset C))$ is an axiom, $S \vdash (A \supset B) \supset ((B \supset C) \supset (A \supset C))$ by L1(a). So (a) by L1(d). (b) By (a) and L1(d). (c) Proof like that of (a). (d) $S \vdash (A \supset B) \supset (\sim B \supset \sim A)$ by L1(ℓ) and L1(a). So (d) by L1(d). (e) Suppose $S \cup \{A\} \vdash B$. Then $S \vdash A \;\mathrm{I}\; B$ by L1(q), and hence $S \cup \{A'\} \vdash A \;\mathrm{I}\; B$ by L1(a). But $(A \;\mathrm{I}\; B) \supset ((A' \supset A) \supset (A' \;\mathrm{I}\; B))$ is valid in the sense of [1]. So $S \cup \{A'\} \vdash (A \;\mathrm{I}\; B) \supset ((A' \supset A) \supset (A' \;\mathrm{I}\; B))$ by the completeness theorem of [1] and L1(a), and hence $S \cup \{A'\} \vdash (A' \supset A) \supset (A' \;\mathrm{I}\; B)$ by L1(d). So, if $S \vdash A' \supset A$, then $S \cup \{A'\} \vdash A' \supset A$ by L1(a), hence $S \cup \{A'\} \vdash A' \;\mathrm{I}\; B$ by L1(d), and hence $S \cup \{A'\} \vdash B$ by L1(c)-(d). (f) Since $(A \vee B) \supset (B \vee A)$ is valid in the sense of [1], $S \vdash (A \vee B) \supset (B \vee A)$ by the completeness theorem of [1] and L1(a). Hence (f) by L1(d). (g)-(ℓ) Proofs like that of (f). (m)-(n) Proofs similar to that of (e). (o)-(p) Proofs similar to that of (f). (q)-(r) By the completeness theorem of [1] and L1(a). (s) Proof similar to that of (e).

L3. (a) If $S \vdash (\forall X)(A \supset B)$, then $S \vdash (\forall X)A \supset (\forall X)B$.

(b) $S \vdash (\forall X')A(X'/X) \supset (\forall X)A$.

(c) If $S \vdash A(\underline{X}/X)$, then $S \vdash (\forall X)A$, so long as $\underline{X}$ is foreign to S and $(\forall X)A$.

(d) $S \vdash (\forall X)(A \supset B) \supset (A \supset (\forall X)B)$.[15]

(e) $S \vdash (\forall X)(A \vee B) \supset (A \vee (\forall X)B)$.[16]

(f) If $S \vdash (\forall X)(A \vee B)$, then $S \vdash A \vee (\forall X)B$, so long as X is foreign to A.

(g) $S \vdash A(\underline{X}/X) \supset (\exists X)A$.

(h) If $S \vdash A(\underline{X}/X) \vee (B(\underline{X}/X) \vee C(\underline{X}/X))$, then $S \vdash (\forall X)A \vee ((\exists X)B \vee (\exists X)C)$, so long as $\underline{X}$ is foreign to S, $(\forall X)A$, $(\exists X)B$, and $(\exists X)C$.

(i) $S \vdash (\forall X)A \supset (\exists X)A$.

(j) $S \vdash ((\exists X)J_k(A) \,\&\, (\forall X)_{i=1}^{\underline{X}} J_i(A)) \supset J_k((\forall X)A)$, for any k from 1 through 3.

(k) $S \vdash (\forall X)\sim J_3(A) \supset (\forall X)(J_1(A) \vee J_2(A))$.

(l) $S \vdash (\forall X)\text{-}J_3(A) \supset (\forall X)\sim J_3(A)$.

(m) $S \vdash \text{-}(\forall X)\text{-}A \;I\; (\exists X)A$.

(n) If $S \vdash A \supset (\exists X)\sim B$, then $S \vdash A \supset \sim(\forall X)B$.

(o) If $S \vdash (\forall X)(\sim\sim A \supset B) \supset C$ then $S \vdash (\forall X)(A \supset B) \supset C$.

(p) If $S \vdash A \supset (B \supset (\forall X)\sim C)$, then $S \vdash A \supset (B \supset \sim(\exists X)C)$.

(q) $S \vdash B \supset (\exists X)(A \supset B)$.

(r) $S \vdash (\exists X)(A \supset B) \supset ((\forall X)A \supset B)$.

(s) $S \vdash ((\exists X)(A \supset B) \supset B) \equiv (((\forall X)A \supset B) \supset B)$.

Proof: (a) Since $(\forall X)(A \supset B) \supset ((\forall X)A \supset (\forall X)B)$ is an axiom, $S \vdash (\forall X)(A \supset B) \supset ((\forall X)A \supset (\forall X)B)$ by L1(a). Hence (a) by L1(d). In case X' and X are the same, (b) by L1(g) and L1(a). So suppose X' and X are distinct, and let $\underline{X}$ be foreign to $(\forall X)A$. $(\forall X')A(X'/X) \supset A(\underline{X}/X)$ $(= (\forall X')A(X'/X) \supset (A(X'/\underline{X})(\underline{X}/X'))$ is an axiom. Hence, by the hypothesis on $\underline{X}$, so is $(\forall X)((\forall X')A(X'/X) \supset A)$. Hence, by L1( ), $S \vdash (\forall X)((\forall X')A(X'/X) \supset A)$. Hence, by (a), $S \vdash (\forall X)(\forall X')A(X'/X) \supset (\forall X)A$. But $(\forall X')A(X'/X) \supset (\forall X)(\forall X')A(X'/X)$ is an axiom. Hence, by L1(a), $S \vdash (\forall X')A(X'/X) \supset (\forall X)(\forall X')A(X'/X)$. Hence (b) by L2(b). (c) See proof of (3.7.12) in [4]. (d) Since $A \supset (\forall X)A$ is an axiom, $S \vdash A \supset (\forall X)A$ by L1(a). Hence $S \vdash ((\forall X)A \supset (\forall X)B) \supset (A \supset (\forall X)B)$ by L2(a). But $(\forall X)(A \supset B) \supset ((\forall X)A \supset (\forall X)B)$ is an axiom. So $S \vdash (\forall X)(A \supset B) \supset ((\forall X)A \supset (\forall X)B)$ by L1(a). So (d) by L2(b). (e) See proof of Lemma 6.7.2 in [5]. (f) Suppose X is foreign to A, in which case $(\forall X)(A \vee B) \supset (A \vee (\forall X)B)$ is well-formed. Then (f) by (e) and L1(d). (g) See proof of Lemma 6.8.5 in [5]. (h) Suppose $S \vdash A(\underline{X}/X) \vee (B(\underline{X}/X) \vee C(\underline{X}/X))$, suppose $\underline{X}$ is foreign to S, $(\forall X)A$, $(\exists X)B$, and $(\exists X)C$, and let X' be new. Then $S \vdash A(\underline{X}/\underline{X}) \vee ((\exists X)B \vee C(\underline{X}/X))$ by (g) and L2(j), hence $S \vdash A(\underline{X}/X) \vee ((\exists X)B \vee (\exists X)C)$ by (g) and L2(k), hence $S \vdash ((\exists X)B \vee (\exists X)C) \vee A(\underline{X}/X)$ by L2(f), hence $S \vdash (\forall X')(((\exists X)B \vee (\exists X)C) \vee A(X'/X))$ by (c), hence $S \vdash ((\exists \underline{X})B \vee (\exists X)C) \vee (\forall X')A(X'/X)$ by (f) and the hypothesis on X', hence $S \vdash ((\exists X)B \vee (\exists X)C) \vee (\forall X)A$ by (b) and L2(j), and hence $S \vdash (\forall X)A \vee ((\exists X)B \vee (\exists X)C)$ by L2(f). (i) Let $\underline{X}$ be an arbitrary individual parameter. Since $(\forall X)A \supset A(\underline{X}/X)$ is an axiom, $S \vdash (\forall X)A \supset A(\underline{X}/X)$ by L1(a). Hence (i) by (g) and L2(b). (j) See

proof of Lemma 6.8.24 in [5]. (k) Let $\underline{X}$ be an individual parameter foreign to $(\forall X)(\sim J_3(A) \supset (J_1(A) \vee J_2(A)))$. By L2(q) $\vdash \sim J_3(A(\underline{X}/X)) \supset (J_1(A(\underline{X}/X) \vee J_2(A(\underline{X}/X)))$. So, by the hypothesis on $\underline{X}$, $\vdash (\forall X)(\sim J_3(A) \supset (J_1(A) \vee J_2(A)))$. So, by L1(a), S $\vdash (\forall X)(\sim J_3(A) \supset (J_1(A) \vee J_2(A)))$. So (k) by (a). (ℓ) Proof like that of (k), but using L2(r) in place of L2(q). (m) See proof of Lemma 6.8.29 in [5]. (n) Let $\underline{X}$ be an individual parameter foreign to $(\forall X)(B \supset \sim\sim B)$. By L1(k) $\vdash B(\underline{X}/X) \supset \sim\sim B(\underline{X}/X)$; hence, by (c) $\vdash (\forall X)(B \supset \sim\sim B)$; hence, by (a), $\vdash (\forall X)B \supset (\forall X)\sim\sim B$; hence, by L1(ℓ), $\vdash (\exists X)\sim B \supset \sim(\forall X)B$; hence by L1(a), S $\vdash (\exists X)\sim B \supset \sim(\forall X)B$; and hence (n) by L2(b). (o)-(p) Proofs similar to that of (n). (q) See proof of Lemma 6.8.10 in [5]. (r) See proof of Lemma 6.8.11 in [5]. (s) See proof of Lemma 6.8.8 in [5].

L4: (a) S $\vdash (\forall X)\text{-}A \supset \text{-}(\exists X)A$.

(b) S $\vdash (\exists X')(A(X'/X) \supset (\forall X)A)$.

(c) If S $\vdash (\forall X)A$, then S $\vdash A(\underline{X}/X)$ for every individual parameter $\underline{X}$ of QC_3.

(d) If S $\vdash \sim A(\underline{X}/X)$ for any individual parameter $\underline{X}$ of QC_3, then S $\vdash \sim(\forall X)A$.

Proof:

(a) Let $\underline{X}$ be foreign to $(\forall X)A$, $(\exists X)B$, and $(\exists X)C$. $J_1(A(\underline{X}/X)) \vee (J_2(A(\underline{X}/X)) \vee J_3(A(\underline{X}/X)))$ is valid in the sense of [1]. So by the completeness theorem of [1]

$$\vdash J_1(A(\underline{X}/X)) \vee (J_2(A(\underline{X}/X)) \vee (J_3(A(\underline{X}/X))$$

so by L3(h) and the hypothesis on $\underline{X}$

$$\vdash (\forall X)J_1(A) \vee ((\exists X)J_2(A) \vee (\exists X)J_3(A)),$$

so by L1(a)

$$\{J_3((\forall X)A), (\forall X)\sim J_3(A)\} \vdash (\forall X)J_1(A) \vee ((\exists X)J_2(A) \vee (\exists X)J_3(A)),$$

so by L2(n)

$$\{J_3((\forall X)A), (\forall X)\sim J_3(A)\} \vdash (\forall X)J_1(A) \vee (\exists X)J_2(A),$$

so by L3(i) and L2(h)

$$\{J_3((\forall X)A), (\forall X)\sim J_3(A)\} \vdash ((\exists X)J_1(A) \,\&\, (\forall X)J_1(A)) \vee (\exists X)J_2(A),$$

so by L3(j) and L2(h)

$$\{J_3((\forall X)A), (\forall X)\sim J_3(A)\} \vdash J_1((\forall X)A) \vee (\exists X)J_2(A),$$

so by L2(m)

$$\{J_3((\forall X)A), (\forall X)\sim J_3(A)\} \vdash J_1((\forall X)A) \vee ((\exists X)J_2(A) \,\&\, (\forall X)\sim J_3(A)),$$ [17]

so by L3(k) and L2(ℓ)

$$\{J_3((\forall X)A), (\forall X)\sim J_3(A)\} \vdash J_1((\forall X)A) \vee ((\exists X)J_2(A) \,\&\, (\forall X)(J_1(A) \vee J_2(A))),$$

so by L3(j) and L2(i)

$$\{J_3((\forall X)A), (\forall X)\sim J_3(A)\} \vdash J_1((\forall X)A) \vee J_2((\forall X)A),$$

so by L2(p)

$$\{J_3((\forall X)A), (\forall X)\sim J_3(A)\} \vdash \sim J_3((\forall X)A),$$

so by L1(c) and L1(r)

$$\{J_3((\forall X)A),(\forall X)\sim J_3(A)\} \vdash -(\forall X)-J_3(A),$$

so by L3(ℓ) and L2(e)

$$\{J_3((\forall X)A),(\forall X)-J_3(A)\} \vdash -(\forall X)-J_3(A),$$

so by L1(q)

$$\{J_3((\forall X)A)\} \vdash (\forall X)-J_3(A) \text{ I } -(\forall X)-J_3(A),$$

so by L2(o)

$$\{J_3((\forall X)A)\} \vdash -(\forall X)-J_3(A),$$

so by L3(m) and L1(d)

$$\{J_3((\forall X)A\} \vdash (\exists X)J_3(A),$$

so by L2(s)

$$\vdash J_3((\forall X)A) \supset (\exists X)J_3(A),$$

so by L3(n)

$$\vdash J_3((\forall X)A) \supset \sim(\forall X)(\sim A \supset A),$$

so by L2(c)

$$\vdash (\forall X)(\sim A \supset A) \supset (\sim(AX)A \supset (\forall X)A),$$

so in particular

$$\vdash (\forall X)(\sim\sim A \supset \sim A) \supset ((\exists X)A \supset (\forall X)\sim A),$$

so by L3(o)

$$\vdash (\forall X)-A \supset ((\exists X)A \supset (\forall X)\sim A),$$

so by L3(p)

$$\vdash (\forall X)-A \supset -(\exists X)A,$$

so by L1(a)

$$S \vdash (AX)-A \supset -(\exists X)A.$$

(b) $(A \supset B) \supset ((B \supset C) \supset (((B \supset A) \equiv (C \supset A)) \supset (C \supset B)))$ is valid in the sense of [1]. So by the completeness theorem of [1]

$$\vdash (A \supset B) \supset ((B \supset C) \supset (((B \supset A) \equiv (C \supset A)) \supset (C \supset B))),$$

so in particular

$$\vdash (B \supset (\exists X)(A \supset B)) \supset (((\exists X)(A \supset B) \supset ((\forall X)A \supset B)) \supset ((((\exists X)(A \supset B) \supset B) \equiv (((\forall X)A \supset B) \supset B)) \supset (((\forall X)A \supset B) \supset (\exists X)(A \supset B)))).$$

But

$$\vdash B \supset (\exists X)(A \supset B),$$

$$\vdash (\exists X)(A \supset B) \supset ((\forall X)A \supset B),$$

and

$$\vdash ((\exists X)(A \supset B) \supset B) \equiv (((\forall X)A \supset B) \supset B)$$

by L3(q), L3(r), and L3(s), respectively. So by L1(d)

$$\vdash ((\forall X)A \supset B) \supset (\exists X)(A \supset B),$$

so in particular

$$\vdash ((\forall X')A(X'/X) \supset (\forall X)A) \supset (\exists X')(A(X'/X) \supset (\forall X)A),$$

so by L3(b) and L1(d)

$$\vdash (\exists X')(A(X'/X) \supset (\forall X)A),$$

so by L1(a)

$$S \vdash (\exists X')(A(X'/X) \supset (\forall X)A).$$ [18]

(c) $(\forall X)A \supset A(\underline{X}/X)$ is an axiom of QC_3. So $S \vdash (\forall X)A \supset A(\underline{X}/X)$ by L1(a). So (c) by L1(d).

(d) $S \vdash (\forall X)A \supset A(\underline{X}/X)$ by the same steps as in (c). So (d) by L2(d).

3. Let S be a set of wffs that is syntactically consistent and infinitely extendible. We extend S into another set S^{ω}, then extend S^{ω} into yet another set S_{ω}, and proceed to show all members of S_{ω} (hence, all members of S) true on a certain truth-value assignment α.

Towards defining S^{ω}, let S^0 be S; and, $(\forall X_n)A_n$ being the alphabetically n-th quantification of QC_3, let S^n be for each n from 1 on $S^{n-1} \cup \{A_n(\underline{X}_n/X_n) \supset (\forall X_n)A_n\}$, where $\underline{X}_n$ is the alphabetically earliest individual parameter of QC_3 foreign to S^{n-1} and $(\forall X_n)A_n$. S^{ω} will then be the union of S^0, S^1, S^2, ...

Towards defining S_{ω}, let S_0 be S^{ω}; and, A_n being the alphabetically n-th wff of QC_3, let S_n be for each n from 1 on $S_{n-1} \cup \{A_n\}$ or S_{n-1} according as $S_{n-1} \cup \{A_n\}$ is syntactically consistent or not. S_{ω} will then be the union of S_0, S_1, S_2, ...

It is easily verified that:

(0) S^{ω} is syntactically consistent,
(1) S_{ω} is syntactically consistent,

and

(2) S_{ω} is maximally consistent.

Proof of (1) is as on p. 328 of [17], but using the syntactic consistency of S^{ω} rather than that of S; and proof of (2) is as on the same page. As for (0), suppose S^n to be syntactically inconsistent, and hence by L1(c) $-(A_n(\underline{X}_n/X_n) \supset (\forall X_n)A_n)$ to be provable from S^{n-1}, and let X'_n be the alphabetically earliest individual variable of QC_3 foreign to $(\forall X_n)A_n$. Then by L3(c) $S^{n-1} \vdash (\forall X'_n) -(A_n(X'_n/X_n) \supset (\forall X_n)A_n)$. But by L4(a)

$$S^{n-1} \vdash (\forall X'_n) -(A_n(X'_n/X_n) \supset (\forall X_n)A_n) \supset -(\exists X'_n)(A_n(X'_n/X_n) \supset (\forall X_n)A_n).$$

So by L1(d)

$$S^{n-1} \vdash -(\exists X'_n)(A_n(X'_n/X_n) \supset (\forall X_n)A_n),$$

i.e.

$S^{n-1} \vdash (\exists X_n')(A_n(X_n'/X_n) \supset (\forall X_n)A_n) \supset \sim(\exists X_n')(A_n(X_n'/X_n) \supset (\forall X_n)A_n)$.
But by L4(b)

$$S^{n-1} \vdash (\exists X_n')(A_n(X_n'/X_n) \supset (\forall X_n)A_n).$$

So by L1(d) S^{n-1} is syntactically inconsistent. So S^n is syntactically consistent if S^{n-1} is. But by assumption S^0 is syntactically consistent. So each one of S^0, S^1, S^2, ..., is syntactically consistent. So, by a familiar argument using L1(a) and L1(b), S^ω is syntactically consistent.

Now let α be the result of assigning to each atomic wff A of QC_3 the truth-value 2. Mathematical induction on the length $\ell(a)$ of an arbitrary wff A of QC_3 will show that:

(i) If $S_\omega \vdash A$, $\alpha(A) = 1$,

(ii) If $S_\omega \vdash \sim A$, $\alpha(A) \neq 3$,

and

(iii) If neither $S_\omega \vdash A$ nor $S_\omega \vdash \sim A$, $\alpha(A) = 2$,

Basis: $\ell(A) = 1$, Proof by the construction of α.

Inductive Step: $\ell(A) > 1$.

Case 1: A is a negation $\sim B$. See Case 1 on p. 328 of [1].

Case 2: A is a conditional $B \supset C$. See Case 2 on pp. 328-9 of [1].

Case 3: A is a quantification $(\forall X)B$. (i) Suppose $S_\omega \vdash (\forall X)B$. Then by L4(e) $S_\omega \vdash B(\underline{X}/X)$ for every individual parameter $\underline{X}$ of QC_3, hence by the hypothesis of the induction $\alpha(B(\underline{X}/X)) = 1$ for every such X, and hence $\alpha((\forall X)B) = 1$. (ii) Suppose $S_\omega \vdash \sim(\forall X)B$, and let $\underline{X}$ be the alphabetically earliest individual parameter of QC_3 such that $B(\underline{X}/X) \supset (\forall X)B$ belongs to S_ω. Then by L1(c) $S_\omega \vdash B(\underline{X}/X) \supset (\forall X)B$, hence by L1(1) and L1(d) $S_\omega \vdash \sim(\forall X)B \supset \sim B(\underline{X}/X$, hence by L1(d) $S_\omega \vdash \sim B(\underline{X}/X)$, hence by the hypothesis of the induction $\alpha(B(\underline{X}/X)) \neq 3$, and hence $\alpha((\forall X)B) \neq 3$. (iii) Suppose neither $S_\omega \vdash (\forall X)B$ nor $S_\omega \vdash \sim(\forall X)B$. If $\alpha(B(\underline{X}/X))$ equaled 3 for any individual parameter $\underline{X}$ of QC_3, then by the hypothesis of the induction $\sim B(\underline{X}/X)$ would be provable from S_ω for that $\underline{X}$, and hence by L4(d) $\sim(\forall X)B$ would be provable from S_ω, against the hypothesis on $\sim(\forall X)B$. If, on the other hand, $(B(\underline{X}/X))$ equaled 1 for every individual parameter $\underline{X}$ of QC_3, then by the hypothesis of the induction $B(\underline{X}/X)$ would be provable from S_ω for every such $\underline{X}$. But $B(\underline{X}/X) \supset (\forall X)B$ is sure to belong to S_ω, and hence by L1(c) to be provable from S_ω, for at least one individual parameter $\underline{X}$ of QC_3. So, if $\alpha(B(\underline{X}/X))$ equaled 1 for every individual parameter $\underline{X}$ of QC_3, then by L1(d) $(\forall X)B$ would be provable from S_ω, against the hypothesis on $(\forall X)B$.

So $\alpha((\forall X)B) = 2$.

Since every member of S belongs to S_ω and hence by L1(c) is provable from S_ω, every member of S is thus sure to evaluate to 1 under α. Hence:

L5. If S is syntactically consistent and infinitely extendible, then S is truth-value verifiable and hence semantically consistent.

Suppose next that S is syntactically consistent but not infinitely extendible; $\underline{X}_i$ being for each i from 1 on the alphabetically i-th individual parameter of QC_3, let M be the mapping on the individual parameters of QC_3 such that $M(\underline{X}_i) = \underline{X}_{2i}$; let M′ be the restriction of M to the individual parameters of QC_3 occurring in S; and let S′ be the M′-rewrite of S. S′ is infinitely extendible, and is easily verified to be syntactically consistent if--as presumed here--S is. So by L5 S′ is truth-value verifiable. But S′ is isomorphic to S. So S is semantically consistent.

So, whether or not S is infinitely extendible,

L6. If S is syntactically consistent, then S is semantically consistent.

So, by the same argument as on p. 329 in [1]:

T1. If S entails A, then $S \vdash A$. (Strong Completeness Theorem for QC_3)

So, taking S to be $\emptyset$,

T2. If A is valid, then $\vdash A$. (Weak Completeness Theorem for QC_3)[19]

REFERENCES

[1] H. Goldberg, H. Leblanc and G. Weaver, " A Strong Completeness Theorem for 3-Valued Logic," Notre Dame Journal of Formal Logic, vol. 15, no. 2, pp. 325-330, 1974.

[2] H. Leblanc, "Truth-Value Semantics for a Logic of Existence," Notre Dame Journal of Formal Logic, vol. 12, no. 2, pp. 153-168, 1971.

[3] H. Leblanc, "Semantic Deviations," Truth, Syntax and Modality, H. Leblanc, ed., North-Holland Publishing Co., Amsterdam, 1973.

[4] H. Leblanc and W. A. Wisdom, Deductive Logic, Allyn & Bacon, Inc., Boston, 1972.

[5] J. B. Rosser and A. R. Turquette, Many-Valued Logics, North-Holland Publishing Co., Amsterdam, 1952.

FOOTNOTES

[1]The paper is a sequel to and presupposes knowledge of [1].

[2]I owe thanks to Professor A. R. Turquette, who suggested the proof of L4(a) below and that of L4(b). I also owe thanks to George Weaver for his counsel and advice throughout the writing of the paper.

[3]The lemma (L5 in the main text) from which our result issues is akin to, though not identical with, Theorem 5.6 in [5].

[4]Our individual variables are in effect what the literature understands by bound individual variables.

[5]Our individual parameters are in effect what the literature understands by free individual variables.

[6]Our predicate parameters are in effect what the literature understands by free predicate variables, and our predicate parameters of degree 0 are what it understands by free sentence variables.

[7]Note that because of (iii) formulas in which identical quantifiers overlap are not counted well-formed.

[8]Following customary practice we shall also write '$\underset{i=1}{\overset{n}{\vee}} A_i$' for '$((\ldots(A_1 \vee A_2) \vee \ldots) \vee A_n)$'.

[9]In [1] we wrote '$\bar{A}$' where we now write '-A'.

[10]With $A \supset (\forall X)A$ presumed to be well-formed, X here is sure to be foreign to A.

[11]In [1] we used 1, 1/2, and 0 as our truth-values, but 1, 2, and 3 prove handier here.

[12]Given the matrices in [1] for ~A and $(A \supset B)$, those for -A, $J_1(A)$, $J_2(A)$, and $J_3(A)$ respectively run:

A	-A	$J_1(A)$	$J_2(A)$	$J_3(A)$
1	3	1	3	3
2	1	3	1	3
3	1	3	3	1

[13]Our interpretation of (∀X)--like that in [5]--is thus of the substitutional sort, and our semantics for QC_2 is of the truth-value sort. For a brief introduction to truth-value semantics, see [3].

[14]Here as in two-valued logic some syntactically consistent sets of wffs are not truth-value verifiable: a case in point is $\{\underline{f}(\underline{x}_1), \underline{f}(\underline{x}_2), \underline{f}(\underline{x}_3), \ldots,$

$\sim(\forall x)\underline{f}(x)\}$, where '$\underline{f}$' is a predicate parameter of degree 1, '$\underline{x}_1$', '$\underline{x}_2$', '$\underline{x}_3$', etc. are all the individual parameters of QC_3, and 'x' is an individual variable. But, as we shall establish below, <u>all</u> syntactically consistent sets of wffs <u>are</u> semantically consistent in the sense just defined. For alternative accounts of semantic consistency in truth-value semantics, see [2].

[15] L3(c)-(d) guarantee that any wff of QC_3 provable by the "axiomatic stipulation" on p. 88 of [5] is provable here, and vice-versa. With $(\forall X)(A \supset B) \supset (A \supset (\forall X)B)$ presumed to be well-formed, X here is sure to be foreign to A.

[16] With $(\forall X)(A \vee B) \supset (A \vee (\forall X)B)$ presumed to be well-formed, X here is sure to be foreign to A.

[17] From this point on the proof of L4(a) is due to Professor Turquette.

[18] The entire proof of L4(b) is due to Professor Turquette.

[19] I owe thanks to Alfred Horn for spotting typos in an early draft of the paper, and in general for his kind and helpful advice.

SUMMARY

A USEFUL FOUR-VALUED LOGIC†

Nuel D. Belnap, Jr.
University of Pittsburgh
USA

Logicians can do a lot of things, but one thing they can do is propose logics to be used. Consider the situation in which an artificial information processor (computer) is receiving information from diverse sources. Then, of course, inconsistency threatens. So it makes sense to consider a sentence as being marked with a T if anyone has said it is true, and as being marked with an F if anyone has said that it is false. Then we are led to the four values: no information, exactly the value T, exactly the value F, and both T and F. What the appropriate significance of the logical connectives "and", "or", and "not" should be in these circumstances is deduced from more than one point of view. In particular, the four elements naturally form an "approximation-lattice" in the sense of Dana Scott, and we may ask, as Scott demonstrates is always appropriate, What are the continuous functions on this lattice? We can also deduce the appropriate connectives by considerations of something like ordinary usage. There then turn out to be interesting connections with relevance logics, which can be exploited in various ways. But throughout the theme is that this is a good logic to use in elaborating an artificial information processing system in the circumstances mentioned above; that is, when inconsistency threatens. For typically, one would not want the system just to go to pieces in the classical way in the presence of having been told that some sentence is both true (by one person) and false (by another).

† This paper will appear in the book Modern Uses of Multiple-Valued Logic.

COMPACTNESS AND ρ-VALUED LOGICS

K.K. Hicken
Michigan State University
U.S.A.

J.M. Plotkin
Michigan State University
U.S.A.

§0. Introduction

One of the most pleasant metatheorems of 2-valued propositional logic is the compactness theorem. The fascination this result holds for mathematicians has a twofold source: "nice" uses of it occur in straight mathematics [see 4, pp. 12-13] and it is related to useful combinatorial lemmas. In [1] Cowen presents a combinatorial lemma from which Rado's lemma and Robinson's valuation lemma follow and he shows their relation to the compactness theorem of 2-valued logic. We refer the reader to [1] for the statements of the lemmas of Rado and Robinson. In [7] Woodruff gives a proof of the compactness of n-valued propositional logics based on Robinson's valuation lemma. In [6] Van Frassen gives a proof for such logics with countably many sentence letters based on König's infinity lemma.

In this paper we study certain notions of compactness for ρ-valued propositional logics where ρ can be infinite and the logics can have infinitary operations. Van Frassen in [6,pp. 179-184] presents a compactness theorem for ρ-valued logics where ρ is a compact topological space and the operators (finitary) of the logic are given continuous interpretations. We assume no topological structure on ρ. Instead we relate our notions of compactness to certain generalizations of Rado's lemma.

We work in Zermelo-Fraenkel set theory with the axiom of choice (ZFC). We do not assume the generalized continuum hypothesis (GCH).

§1. Notation and basic definitions.

We identify an ordinal with its set of predecessors and a cardinal with the smallest ordinal having that cardinality. The letters α, ξ are used for ordinals and the letters $\kappa, \lambda, \rho, \mu, \theta$ for cardinals. ω has its usual meaning. For S a set $|S|$ denotes the cardinality of S, $P(S) = \{X | X \subseteq S\}$, $P_\kappa(S) = \{X | X \subseteq S \text{ and } |X| < \kappa\}$. For R,S sets $S_R = \{f | f: S \to R\}$. $\lambda^{<\kappa} = \sum_{\mu<\kappa} \lambda^\mu$. κ^+ denotes the first cardinal greater than κ. κ is a limit cardinal if $\kappa \neq \lambda^+$ for all λ. $cf(\alpha)$ is the smallest ξ which can be mapped onto a cofinal subset of α. In the following definitions κ is infinite. κ is regular if $cf(\kappa) = \kappa$. κ is weakly inaccessible if it is both a limit cardinal and regular. κ is inaccessible if it is regular and for each $\lambda < \kappa$, $\mu < \kappa$ we have $\lambda^\mu < \kappa$.

ω is inaccessible. It is not known if the existence of uncountable (weakly) inaccessibles is consistent with ZFC (though not many doubt that such is the case). In the presence of GCH weakly inaccessible = inaccessible. We refer the reader to [5] for further details on the above definitions.

Definition $\omega \leq \varkappa \leq \lambda$. An $L(\varkappa,\lambda,\rho)$ logic is a quadruple $\langle \mathscr{A}, \mathcal{O}, M, D \rangle$ where

i) $\mathscr{A}$ is a set of sentence letters, $|\mathscr{A}| \leq \lambda$,

ii) $\mathcal{O}$ is a set of operation symbols, $|\mathcal{O}| \leq \lambda$, each element $O \in \mathcal{O}$ has a unique ordinal degree $\xi < \varkappa$ (we write O^{ξ} to indicate ξ is the degree of O).

iii) For each $O^{\xi} \in \mathcal{O}$ M contains a mapping $O_M : {}^{\xi}\rho \to \rho$, O_M is called the matrix of O.

iv) D is a subset of ρ, the elements of D are called the designated values of the logic.

Let L be an $L(\varkappa,\lambda,\rho)$ logic. The set W of sentences of L is the smallest set of expressions containing $\mathscr{A}$ and such that if $\{W_\alpha | \alpha < \xi\} \subseteq W$ and $O^{\xi} \in \mathcal{O}$ then $O\{W_\alpha | \alpha < \xi\} \in W$. A mapping $f : A \to \rho$ where $A \subseteq \mathscr{A}$ is called a partial valuation of L. If $A = \mathscr{A}$ we say f is a valuation. A mapping $h : W \to \rho$ is called an interpretation of L if the following condition is satisfied: if $\{W_\alpha | \alpha < \xi\} \subseteq W$ and $O^{\xi} \in \mathcal{O}$ then $h(O\{W_\alpha | \alpha < \xi\}) = O_M\{h(W_\alpha) | \alpha < \xi\}$. Every valuation of L can be uniquely extended to an interpretation of L. We identify a valuation with its extension to W.

Definition. $\Sigma \subseteq W$. Σ is D-satisfiable if there is a valuation f of L such that $f(\sigma) \in D$ for all $\sigma \in \Sigma$.

Definition. L is θ-compact if for each $\Sigma \subseteq W$ the following holds: if each $\Gamma \in P_\theta(\Sigma)$ is D-satisfiable then Σ is D-satisfiable.

Remarks. ω-compact is just called compact. If L is θ-compact, it is μ-compact for all $\mu \geq \theta$.

We shall study $\varkappa$-compactness of $L(\varkappa,\lambda,\rho)$ logics. First we give an example of an $L(\omega,\omega,\omega)$ logic which is not ω-compact.

Example. Let $\mathscr{A} = \{p_i | i < \omega\}$. Let $\mathcal{O} = \{O_n | n < \omega\}$ where the degree of O_n is n and the matrix of O_n is defined as follows:

$$(O_n)_M(i_1,\cdots,i_n) = \begin{cases} 0 & \text{if } i_1 < i_2 < \cdots < i_n \\ 1 & \text{otherwise} \end{cases}$$

D is $\{0\} \subseteq \omega$. Consider Σ where

$$\Sigma = \{O_n \langle p_1, p_2, \cdots, p_{n-1}, p_0 \rangle | n < \omega\}.$$

Each finite subset is D-satisfiable. But Σ is not D-satisfiable. A valuation that would make Σ D-satisfiable would have to assign p_0 a value greater than any $i < \omega$.

Remark. A similar example due to R. Meyer which uses only one degree two operation symbol appears in [6,pp. 144].

We now give a series of definitions which enable us to describe a generalization of Rado's lemma.

Definition. $C \subseteq P(S)$. C is a κ-cover of S if for each $X \in P_\kappa(S)$ there is an $A \in C$ with $X \subseteq A$.

Definition. C a κ-cover of S. R a set. A collection of mappings $\mathcal{J} = \{f_A | f_A : A \to R, A \in C\}$ is called a C-R valuation.

Definition. $\omega \leq \kappa \leq \lambda$. κ, λ, ρ have the Rado property (written $[\kappa, \lambda, \rho]$) if whenever S is a set with $|S| \leq \lambda$, C is a κ-cover of S, and $\mathcal{J}$ is a C-R valuation where $|R| \leq \rho$, there exists an $f : S \to R$ such that for each $X \in P_\kappa(S)$ there is an $A \in C$ with $X \subseteq A$ and $f \restriction X = f_A \restriction X$.

Such an f need not be unique. But we abuse notation by writing $f = \lim_A f_A$.

This generalization of Rado's lemma (in slightly different form) was one of two such generalizations given in [2]. Earlier Jech had defined what we call $[\kappa, \lambda, 2]$ and discussed its relation to compactness properties of certain infinitary first order languages [3].

From [2] we have the following:

Proposition 1. If $[\kappa, \lambda, 2]$ then κ is weakly inaccessible.

Proposition 2. If $[\kappa, \lambda, 2]$ then for all $\rho < \kappa$ $[\kappa, \lambda, \rho]$.

For $\rho < \omega$ $[\omega, \lambda, \rho]$ is provable in ZFC. It is just a version of Rado's lemma.

§2. κ-compactness of $L(\kappa, \lambda, \rho)$ logics

Theorem 1. If $\lambda^{(\kappa)} = \lambda$ then $[\kappa, \lambda, \rho]$ implies every $L(\kappa, \lambda, \rho)$ logic is κ-compact.

Proof. Let L be an $L(\kappa, \lambda, \rho)$ logic. To avoid trivialities we assume $2 \leq \rho$. Since $[\kappa, \lambda, \rho]$ holds, proposition 1 implies κ is regular.

The regularity of κ together with $\lambda^{<\kappa} = \lambda$ yields $|W| \leq \lambda$. Let $\Sigma \subseteq W$ be such that $\Gamma \in P_\kappa(\Sigma)$ is D-satisfiable. For each Γ choose an interpretation $f_\Gamma : W \to \rho$ such that $f_\Gamma(\gamma) \in D$ for all $\gamma \in \Gamma$. $P_\kappa(\Sigma)$ is a κ-cover of Σ. Let $f = \lim_\Gamma f_\Gamma$. It is easy to show that f is an interpretation of L and $f(\sigma) \in D$ for all $\sigma \in \Sigma$.

Theorem 2. If $2 \leq \rho < \kappa$ and κ is inaccessible then every $L(\kappa,\lambda,\rho)$ logic is κ-compact implies $[\kappa,\lambda,\rho]$.

Proof. Let S be a set with $|S| \leq \lambda$, C a κ-cover of S, R a set with $|R| \leq \rho$ and $\mathcal{J}$ a C-R valuation. Without loss of generality Proposition 2 allows us to assume $R = \rho = \{0,1\}$.

Let us consider the following $L(\kappa,\lambda,\rho)$ logic $\langle \mathscr{A}, \mathcal{O}, M, D\rangle$: $\mathscr{A} = S$, $D = \{0\} \subseteq \{0,1\}$, $\mathcal{O}$ has a degree one operation symbol $\neg$ and for each $\xi < \kappa$ it has operation symbols $\wedge^\xi$, $\vee^\xi$. These operation symbols have the following matrices:

$$\neg_M\{r\} = \begin{cases} 0 & \text{if } r \neq 0 \\ 1 & \text{otherwise} \end{cases}$$

$$\wedge^\xi_M\{r_\alpha \mid \alpha < \xi\} = \begin{cases} 0 & \text{if } r_\alpha = 0 \text{ for all } \alpha < \xi \\ 1 & \text{otherwise} \end{cases}$$

$$\vee^\xi_M\{r_\alpha \mid \alpha < \xi\} = \begin{cases} 0 & \text{if } r_\alpha = 0 \text{ for some } \alpha < \xi \\ 1 & \text{otherwise} \end{cases}$$

For $F \in P_\kappa(S)$ let $\widetilde{F} = \{h \mid h = f_A \restriction F, F \subseteq A \in C\}$. $|\widetilde{F}| \leq \rho^\theta$ where $|F| = \theta < \kappa$. Since κ is inaccessible and $\rho < \kappa$, $|\widetilde{F}| < \kappa$. For each $h \in \widetilde{F}$ let φ_h be the sentence

$$\wedge^{\xi_1}\{s \mid s \in F \ \&\ h(s) = 0\} \wedge^2 \wedge^{\xi_2}\{\neg s \mid s \in F \ \&\ h(s) = 1\}.$$

If f is a valuation on L then $f(\varphi_h) = 0$ iff $f \restriction F = h$. Now let φ_F be the sentence

$$\vee^\xi\{\varphi_h \mid h \in \widetilde{F}\}.$$

Finally let $\Sigma = \{\varphi_F \mid F \in P_\kappa(S)\}$. Let $\Gamma \in P_\kappa(\Sigma)$. $\Gamma = \{\varphi_{F_\xi} \mid \xi < \theta < \kappa\}$. Since κ is regular, there is an $A \in C$ such that $\bigcup_{\xi<\theta} F_\xi \subseteq A$. Let

$h_\xi = f_A \restriction F_\xi$. φ_{h_ξ} is in φ_{F_ξ}. We extend the partial valuation f_A to a valuation g on L (in some arbitrary way). It is easy to see that $g(\varphi_{h_\xi}) = 0$ and hence that $g(\varphi_{F_\xi}) = 0$. Thus Γ is D-satisfiable.

By the κ-compactness of $L(\kappa,\lambda,\rho)$ logics there is a valuation $f:S \to \rho$ such that $f(\sigma) = 0$ for all $\sigma \in \Sigma$. We claim $f = \lim_A f_A$. Let $F \in P_\kappa(S)$. For some φ_h in φ_F we have $f(\varphi_h) = 0$. Therefore $f \restriction F \in \tilde{F}$; i.e., there is an $A \in C$ such that $f_A \restriction F = f \restriction F$.

We now combine theorems 1 and 2.

Theorem 3. If $\lambda^{<\kappa} = \lambda$, κ inaccessible, and $2 \leq \rho < \kappa$, then $[\kappa,\lambda,\rho]$ iff every $L(\kappa,\lambda,\rho)$ logic is κ-compact.

Remark. Theorem 3 should be compared with 2.2 of [3].

Consider now the special case $\kappa = \omega$. ω is inaccessible and $\lambda^{<\omega} = \lambda$ for all $\lambda \geq \omega$.

Corollary 1. i) $2 \leq \rho < \omega$, $\omega \leq \lambda$ $[\omega,\lambda,\rho]$ iff every $L(\omega,\lambda,\rho)$ logic is compact. ii) the compactness of 2-valued propositional logic implies the compactness of n-valued propositional logic for each $n < \omega$.

Remarks. A reasonable question to ask in the light of Theorem 1 is "when does $[\kappa,\lambda,\rho]$ hold?" From Proposition 1 we know κ must necessarily be weakly inaccessible. Thus for $\omega < \kappa$, $[\kappa,\lambda,\rho]$ is a "large cardinal" property. In [3] it is shown that if $\lambda^{<\kappa} = \lambda$, κ inaccessible, and κ is λ-compact (defined in [3,pp.173]) then $[\kappa,\lambda,2]$, and hence $[\kappa,\lambda,\rho]$ for all $\rho < \kappa$ by Proposition 2.

It appears that for $\omega < \kappa$ some very strong assumptions are needed to insure κ-compactness of $L(\kappa,\lambda,\rho)$ logics. Of course κ-compactness is weaker than compactness. In particular our example L of an $L(\omega,\omega,\omega)$ logic which is not compact when viewed as an $L(\omega_1,\omega_1,\omega)$ logic is ω_1-compact.

REFERENCES

1. Cowen, R.H., Some Combinatorial Theorems Equivalent to the Prime Ideal Theorem, Proc. AMS, 41 (1973), 268-273.
2. Hickin, K.K. and Plotkin, J.M., Some Algebraic Properties of Weakly Compact and Compact Cardinals, preprint.

3. Jech, T.J., Some Combinatorial Problems Concerning Uncountable Cardinals, Annals of Math. Logic, 5 (1973), 165-198.

4. Kreisel, G. and Krivine, J.L., Elements of Mathematical Logic, North Holland, 1967.

5. Kuratowski, K. and Mostowski, A., Set Theory, North Holland, 1968.

6. Van Frassen, B.C., Formal Semantics and Logic, Macmillan, New York, 1971.

7. Woodruff, P.W., On Compactness in Many Valued Logic I, Notre Dame J. of Formal Logic, 3 (1973), 405-407.

FINITELY-MANY-VALUED LOGICS WITH INFINITELY-MANY-VALUED EXTENSIONS: TWO EXAMPLES

Dolph Ulrich[1]
Purdue University
U.S.A.

Proofs that certain logics fail to be finitely-many-valued[2] first appeared in Gödel's [6], concerned with the intuitionistic sentential calculus, H, and in Dugundji's [3], devoted to the Lewis systems S1-S5 and S6. In [11] the present author posed a general question: can such results always be extended to the class of all (nonempty) subsystems of the system in question? It was established in [11] that the answer is affirmative in the case of the negation-free fragment of H, and in [14] the weak implicational calculus of Church's [2] was treated similarly. Related, though less complete, results have been obtained by Pahi [8], who shows that a wide class of implicational calculi in the vicinity of the implicational fragment of H fail to be finitely-many-valued, and by Anderson [1], who establishes the nonexistence of finitely-valued calculi lying between H and the calculus KC of [4].[3] So far as the author knows, however, the general question has remained open. After some preliminary extensions of one of Dugundji's results in §1 below, we will show that the answer is negative by constructing a 3-valued subsystem of S6. Investigation of the general question thus gives way to consideration of the family of questions of the form do all subsystems of X fail to be finitely-many-valued? (where 'X' ranges over systems not themselves finitely-valued). Chief among the open questions of this form seems to be the one where X = H; this question is answered, also negatively, in §2.

§1. Confirmation of a conjecture of Prior's, and some results concerning subsystems of S6.

Conserving space, we refer the reader to [10] (p. 311) or [7] (p. 269) for the formulation of S6, though we adopt the prefix notation of the former, and to [7] (pp. 281-283) for the corresponding semantic treatment. Our terminology concerning matrices, which we also presume is familiar, is available in [12].

Let S be the set of well-formed formulas of S6, with W the set of nonnegative integers and R the relation each $i > 0$ bears to itself, its predecessor and 0. Let V map $S \times W$ into $\{T,F\}$ in such a way that $V(p,1) = T$, $V(p,i) = F$ for $i \neq 1$, and for all $\alpha,\beta \in S$ and $i \in W$, $V(N\alpha,i) = T$ iff $V(\alpha,i) = F$, $V(K\alpha\beta,i) = T$ iff $V(\alpha,i) = V(\beta,i) = T$, and $V(M\alpha,i) = T$ iff $i = 0$ or $V(\alpha,j) = T$ for some j such that iRj. Then $\langle W,R,V\rangle$ is an S6-model in the sense of [7] (p. 281) so--as can also be easily verified directly--$V(\tau,i) = T$ for each theorem τ of S6 and $i > 0$. Writing, as is customary, $C\alpha\beta$ for $NK\alpha N\beta$ and L for NMN, we note that for all $\alpha,\beta \in S$ and $i \in W$, $V(C\alpha\beta,i) = T$ iff $V(\alpha,i) = F$ or $V(\beta,i) = T$ and $V(L\alpha,i) = T$ iff $i \neq 0$ and $V(\alpha,j) = T$ for every j such that iRj. Moreover, $V(L\alpha,0) = F$ and $V(M\alpha,0) = T$ for each $\alpha \in S$.

Prior ([9], p. 125) remarks that there are but finitely many nonequivalent modalities in S7 and so in S8, but conjectures that there are infinitely many in S6. To establish this claim, which will allow us to generalize Dugundji's result for S6, we consider the sequence of wffs of S6 in which $s_1 = p$ and, for each positive integer n, $s_{2n} = Ms_{2n-1}$ and $s_{2n+1} = Ls_{2n}$.

LEMMA 1. $V(s_k,1) = T$ for each k.

Proof. $V(s_1,1) = T$ by stipulation, so assume on inductive hypothesis that $V(s_m,1) = T$. Then $V(s_{m+1},1) = T$ as well: if m is odd, then s_{m+1} is Ms_m and the result follows from the induction hypothesis and the fact that 1 bears R to itself; if m is even, s_{m+1} is Ls_m and the result follows because $V(s_m,1) = V(s_m,0) = T$ and 1 bears R only to itself and 0.

LEMMA 2. For each integer $j > 1$, if $V(s_k,j) = F$ when $1 \leq k \leq 2j - 3$ and $V(s_k,j) = T$ for $k > 2j - 3$, then $V(s_k,j + 1) = F$ when $1 \leq k \leq 2j - 1$ and $V(s_k,j + 1) = T$ for $k > 2j - 1$.

Proof. Pick $j > 1$ and say $V(s_1,j) = \ldots = V(s_{2j-3},j) = F$ but $V(s_{2j-2},j) = V(s_{2j-1},j) = \ldots = T$.
Of course we have stipulated that $V(s_1,j + 1) = F$, so say $1 \leq k < 2j - 1$ and assume that $V(s_k,j + 1) = F$. If k is even, $s_{k+1} = Ls_k$ and $V(s_{k+1},j + 1) = F$ because $V(s_k,j + 1) = F$ and $j + 1$ bears R to itself. If k is odd, $s_{k+1} = Ms_k$ and $V(s_{k+1},j + 1) = F$ also, this time because $V(s_k,j + 1) = V(s_k,j) = V(s_k,0) = F$ and $j + 1$ bears R only to itself, j and 0.

Next consider $s_{2j} = Ms_{2j-1}$. We have $V(s_{2j-1},j) = T$ from the hypothesis of the the lemma; since $j + 1$ bears R to j, then, $V(s_{2j},j + 1) = T$ as well. Assume, now, on inductive hypothesis, that $V(s_{2j+m},j + 1) = T$. Then clearly $V(s_{2j+m+1},j + 1) = T$ also: if $2j + m + 1$ is even, $s_{2j+m+1} = Ms_{2j+m}$ and the result follows from the induction hypothesis and the fact that $j + 1$ bears R to itself; if $2j + m + 1$ is odd, $s_{2j+m+1} = Ls_{2j+m}$ and the result follows from the induction hypothesis and the additional facts that $V(s_{2j+m},j) = V(s_{2j+m},0) = T$ and $j + 1$ bears R only to itself, j and 0.

LEMMA 3. For each positive integer j, $V(s_k,j) = F$ for all s_k (if any) such that $k \leq 2j - 3$ and $V(s_k,j) = T$ for $k > 2j - 3$.

The proof is now immediate, by induction on j: lemma 1 constitutes the base case, and lemma 2 the induction step.

With these inductive doings behind us, we are ready for a number of results.

THEOREM 1 (Prior's Conjecture). S6 has infinitely many nonequivalent modalities.

Proof. For $i < j$, $V(s_{2j},j + 1) = T$ but $V(s_{2i},j + 1) = F$ by lemma 3, so $V(Cs_{2j}s_{2i},j + 1) = F$; thus no two of s_2, s_4, s_6, ... are equivalent in S6.

COROLLARY 1 (Dugundji's Result). S6 is not finitely-many-valued.

Proof. Let $\mathfrak{M}$ be any m-valued model of S6. Then for some $i < j \leq m^m + 1$, s_{2i} and s_{2j} must receive identical values for each assignment of values of $\mathfrak{M}$ to p. Since Cpp (being a theorem of S6) is an $\mathfrak{M}$-tautology, then, $Cs_{2j}s_{2i}$ must also be an $\mathfrak{M}$-tautology, so that $\mathfrak{M}$ fails to be characteristic for S6.

To improve Dugundji's result, we need an additional lemma:

LEMMA 4. Let ϕ be any wff of S6 in which no modal operators occur. Then for all wffs α and β, there exists a wff $\phi(\alpha,\beta)$ which results from ϕ by replacing zero or more occurrences of certain letters with α and the remaining occurrences of letters (if any) with β for which we have $\vdash_{S6} C\phi(\alpha,\beta)C\alpha\beta$.

Proof. Let ψ result from ϕ by replacing the first occurrence of a sentential letter in ψ with p, the second with q, and so on. Then ψ is not provable in the classical sentential calculus, SC, since no letter occurs twice in ψ. Consequently, there exists an assignment of T's and F's to the letters occurring in ψ on which ψ takes, according to the usual two-valued truth-tables for SC, the value F. If $\phi(\alpha,\beta)$ is now obtained from ψ by replacing each letter thus assigned a T with α and each letter thus assigned an F with β, $\phi(\alpha,\beta)$ will be as required for the lemma.

Counting as a <u>subsystem</u> of S6 any sentential logic with the same language whose theorems are provable in S6, we have:

COROLLARY 2. No subsystem of S6 whose theorems include at least one wff free of modal operators is finitely-many-valued.

Proof. Let T be any such system, with ϕ such a theorem, and suppose T has an m-valued, characteristic matrix, $\mathfrak{M}$. Then the set of theorems of T must be closed under substitution, so we may assume without loss of generality that the only sentential letter occurring in ϕ is p. As in the proof of corollary 1, for some $i < j \leq m^m + 1$, s_{2i} and s_{2j} receive identical values for each assignment of values of $\mathfrak{M}$ to p. Since ϕ is an $\mathfrak{M}$-tautology, then, so must be any formula obtained from ϕ by replacing zero or more occurrences of p with s_{2j} and the rest (if any) with s_{2i}. In particular, $\phi(s_{2j},s_{2i})$ must be an $\mathfrak{M}$-tautology, where the latter is chosen--as lemma 4 assures it can be--so that $\vdash_{S6} C\phi(s_{2j},s_{2i})Cs_{2j}s_{2i}$. But then $\phi(s_{2j},s_{2i})$ must be provable in T and so in S6 as well, whence $\vdash_{S6} Cs_{2j}s_{2i}$, contrary to our earlier result.

Of course corollary 2 can be improved--a similar argument shows, for example, that no subsystem of S6 whose theorems include at least one wff of the form $L\alpha$ with α free of modal operators is finitely-valued. It cannot, however, be extended to <u>all</u> nonempty subsystems of S6:

THEOREM 2. Let S* be the subsystem of S6 whose theorems are the substitution instances of MMp and whose rules of inference are substitution and detachment.[4] Then the theorems of S* are precisely the tautologies of the following 3-valued truth-table (3 is the sole designated value):

K	1	2	3	N	M
1	1	1	1	1	2
2	1	1	1	1	3
3	1	1	1	1	3

Proof. Obviously each theorem of S* is a tautology: for each wff α, Mα can take only the value 2 or the value 3, and in either case MMα accordingly takes the value 3. Obviously also, all sentential letters, conjunctions and negations fail to be tautologies for they take the value 1 when each letter occurring in them is assigned that value. The only possible tautologous nontheorems, then, are those wffs of the form Mα where α itself is a letter, a conjunction or a negation--and since, in these cases, α takes the value 1 when each letter is assigned a 1, Mα here takes the value 2.

2. A finitely-many-valued subsystem of H.

S6 is such a peculiar system that one might perhaps have expected something like theorem 2. It seems more surprising that a similar result holds for the intuitionistic calculus, H. But let G be the subsystem of H whose theorems are precisely those wffs of the form NNα, where α is a theorem of the classical sentential calculus, SC.[5] (G's rules may be taken to be those of H, substitution and detachment, though the latter can be dropped without diminishing G's stock of theorems.) Then we have:

THEOREM 3. G is finitely-many-valued.

Before turning to the proof proper, we attend to some preliminaries. We first extract a definition and modify a lemma from [12], familiarity with which must here be presumed.

DEFINITION. Where P is a sentential calculus and α and β are wffs, α and β are interchangeable in P (for short, $\alpha \simeq_P \beta$) just in case replacement of one or more occurrences of α with occurrences of β in any theorem (respectively, nontheorem) of P results in a theorem (respectively, nontheorem) of P.

Clearly, $\simeq_P$ is always an equivalence relation.

LEMMA 5. Let S_1 be the set of wffs of G in which the only sentential letter occurring is p. Then if (a) $\simeq_G$ partitions S_1 into but finitely many equivalence classes and (b) each nontheorem of G has a substitution instance in S_1 not provable in G, then G is finitely-many-valued.

The proof is an easy modification of that presented for lemma 2 of [12]: if (a) holds then the Lindenbaum matrix $Ld^{\{P\}}(G)$ of [12] (p. 364), in which the truth-values are taken to be the equivalence classes in question, is clearly finite; and if (b) holds as well, then its tautologies are precisely the theorems of G.

Of course no special features of G were employed in this proof, and lemma 5 accordingly applies to arbitrary sentential calculi. The proof of the following lemma, however, depends heavily on just such special features of G.

LEMMA 6. Let α be any member of S_1. Then:

i. if $\alpha \simeq_{SC}$ Cpp and α does not begin with NN, $\alpha \simeq_G$ Cpp;
ii. if $\alpha \simeq_{SC}$ Cpp and α begins with NN, $\alpha \simeq_G$ NNCpp;
iii. if $\alpha \simeq_{SC}$ p, $\alpha \simeq_G$ p;
iv. if $\alpha \simeq_{SC}$ Np, $\alpha \simeq_G$ Np;
v. if $\alpha \simeq_{SC}$ KpNp and α does not begin with N, $\alpha \simeq_G$ KpNp;
vi. if $\alpha \simeq_{SC}$ KpNp and α begins with N, $\alpha \simeq_G$ NCpp.

Proof. Ad (i). Let ϕ be any wff of G containing α as a subformula, and let $\phi(Cpp)$ be any wff resulting from ϕ by replacing one or more occurrences of α with occurrences of Cpp. Assume, first, that ϕ is a theorem of G. Then $\phi = NN\psi$ for some theorem ψ of SC. Since α does not begin with NN, α must be a subformula of $N\psi$; and since the latter cannot be provable in SC, α must in fact be a subformula of ψ. But $\alpha \simeq_{SC} Cpp$, so any wff $\psi(Cpp)$ resulting from ψ by replacing one or more occurrences of α with occurrences of Cpp must also be provable in SC. Consequently, $NN\psi(Cpp)$ is provable in G for each such wff $\psi(Cpp)$. In particular, $\phi(Cpp)$ is a theorem of G. An analogous argument leads from the assumption that $\phi(Cpp)$ is provable in G to the conclusion that ϕ is also.

Ad (ii). The proof is similar, noting this time that if α is a subformula of $NN\psi$ and begins with NN, then α must be either a subformula of ψ or else $NN\psi$ itself.

Ad (iii)-(vi). The proofs here are somewhat similar to those indicated above; for brevity's sake, they are left to the reader.

We are, finally, prepared to establish theorem 3 by showing that clauses (a) and (b) of lemma 5 are both satisfied.

Proof of (a). Trivially, each member of S_1 is interchangeable in SC with Cpp, p, Np or KpNp. By lemma 6, then, each member of S_1 is interchangeable in G with Cpp, NNCpp, p, Np, KpNp or NCpp. So $\simeq_G$ partitions S_1 into finitely many equivalence classes.

Proof of (b). Let ϕ be any formula not provable in G. Then either ϕ fails to be of the form $NN\psi$, in which case the result of replacing each letter in ϕ with p is a substitution instance of ϕ not provable in G either, or else ϕ is of the form $NN\psi$ for some formula ψ not provable in SC. In the latter case (compare the proof of lemma 4 above) there exists some assignment of T's and F's to the letters occurring in ψ on which ψ takes, according to the usual two-valued truth-tables, the value F. Then if ψ' results from ψ by replacing each letter thus assigned a T with Cpp and each letter thus assigned an F with NCpp, $NN\psi'$ is a substitution instance of ϕ not provable in G, and our proof is complete.

F O O T N O T E S

1. Some of the results established here were announced in [13]. The author would also like to thank the referee for several suggestions.

2. A logic is <u>finitely-many-valued</u> if and only if there is a truth-table with a finite number of truth-values whose tautologies are precisely the theorems of the logic in question, that is, if and only if the logic has a finite characteristic matrix in the sense of [12].

3. Actually, this result can be improved, for Anderson shows (theorem 6.3) that KC and H coincide in their negation-free fragments. From the result of [11], then, it follows that in fact no subsystem of KC whose theorems include even one negation-free formula can be finitely-many-valued.

4. S6 is usually formulated with an additional rule, adjunction. If S** is the subsystem of S6 with MMp as axiom and substitution, detachment and adjunction as rules,

then S** is also finitely-many-valued: simply alter the matrix of theorem 2 by letting K33 = 3.

5. That each theorem of G is a theorem of H, so that G is indeed a subsystem of H, amounts to the familiar result that the double negation of each theorem of SC is provable in H. The latter appears to have been first established in [5].

REFERENCES

[1] J.G. Anderson, "Superconstructive Propositional Calculi with Extra Axiom Schemes Containing One Variable", Z. Math. Logik Grundlagen Math., vol. 18, pp. 113-130, 1972.

[2] A. Church, "The Weak Theory of Implication", Kontrolliertes Denken, Munich, 1951.

[3] J. Dugundji, "Note on a Property of Matrices for Lewis and Langford's Calculi of Propositions", J. Symbolic Logic, vol. 5, pp. 150-151, 1940.

[4] M.A.E. Dummett and E.J. Lemmon, "Modal Logics between S4 and S5", Z. Math. Logik Grundlagen Math., vol. 5, pp. 250-264, 1959.

[5] V. Glivenko, "Sur Quelques Points de la Logique de M. Brouwer", Acad. Belg., Bull. des Sci., ser. 5, vol. 15, pp. 183-188, 1929.

[6] K. Gödel, "Zum intuitionistischen Aussagenkalkül", Ergebnisse eines mathematischen Kolloquiums, vol. 4, p. 40, for 1931-1932, pub. 1933.

[7] G.E. Hughes and M.J. Cresswell, An Introduction to Modal Logic, Methuen, London, 1968.

[8] B. Pahi, "A Method for Proving the Non-existence of Finite Characteristic Models for Implicational Calculi", Z. Math. Logik Grundlagen Math., vol. 18, pp. 169-172, 1972.

[9] A.N. Prior, Time and Modality, Clarendon Press, Oxford, 1957.

[10] A.N. Prior, Formal Logic, Clarendon Press, Oxford, 1962.

[11] D. Ulrich, "Subsystems of Full Positive Logic", abstract, J. Symbolic Logic, vol. 36, p. 696, 1971.

[12] D. Ulrich, "Some Results Concerning Finite Models for Sentential Calculi", Notre Dame J. Formal Logic, vol. 13, pp. 363-368, 1972.

[13] D. Ulrich, "Subsystems of S6, and a Conjecture of Prior's", abstract, J. Symbolic Logic, vol. 39, p. 200, 1974.

[14] D. Ulrich, "The Nonexistence of Finite Characteristic Matrices for Subsystems of R_I", Proceedings of the International Conference on Relevance Logics, to appear.

TRUTH FUNCTIONALITY AND NATURAL DEDUCTION

James D. McCawley
Department of Linguistics
University of Chicago

This study is concerned with the question of the extent to which a well-known system of rules of inference for propositional logic, namely that of Thomason [3], forces the connectives to be truth-functional.

For a given formal language L, let V(L) consist of all functions which associate a value T or F to each proposition of L, i.e., $V(L) = \{T, F\}^L$. The members of V(L) are called valuations on L. For a given system of rules of inference R, let V(L, R) be the set of all valuations for which R leads only to true conclusions when applied to true premises, i.e., the set of all f in V(L) such that if $f(A_1) = \ldots = f(A_n) = T$ and R authorizes the inference from $\{A_1, \ldots, A_n\}$ to B, then $f(B) = T$. A connective \$ of L is truth-functional relative to a given set of valuations $V \subseteq V(L)$ if for every f, $g \in V$, if $f(A_1) = g(B_1), \ldots, f(A_n) = g(B_n)$, then $f(\$A_1 \ldots A_n) = g(\$B_1 \ldots B_n)$. To say that a set of rules of inference R forces the connectives of L to be truth-functional is to say that all of the connectives of L are truth-functional relative to V(L, R). I note in passing that a set of rules of inference sometimes forces the connectives of a language to be truth-functional in a trivial way. For example, if we take the system suggested facetiously by Prior [2], which contains a connective 'tonk' with the rules of inference

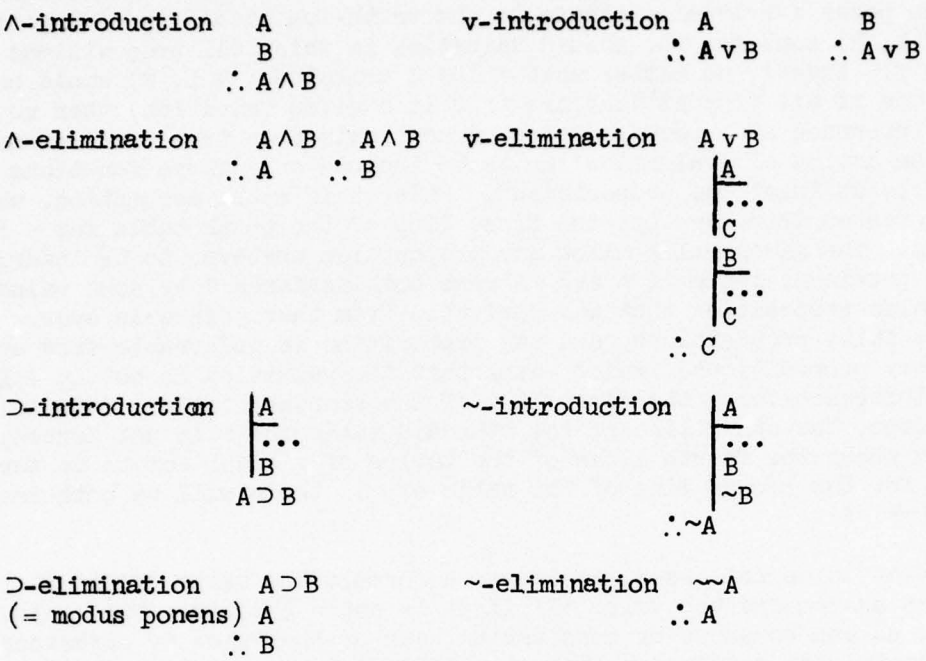
$$\frac{A}{A \text{ tonk } B} \qquad \frac{A \text{ tonk } B}{B}$$

then V(L, R) contains at most two functions: the one which assigns to all propositions the value T and the one which assigns to all propositions the value F (proof: if V(L, R) contained a function f which was neither identically T nor identically F, then there would be propositions A and B such that f(A) = T and f(B) = F, and B would be inferrable from A via the above rules of inference, via the intermediate step of 'A tonk B'; but this contradicts the assumption that $f \in V(L, R)$). In that case, any connective is truth-functional relative to V(L, R). If these two valuations are ruled out of consideration as absurd, then V(L, R) is empty.

Truth-functionality is a property of a connective only in relation to a class of valuations. A connective may be truth-functional with respect to one class V_1 of valuations but non-truth-functional with respect to a larger class V_2. For example, V_1 might be the class of all 'standard' valuations, that is, those that conform to the truth tables that are to be found in the most popular logic textbooks[1], and V_2 might contain not only the standard valuations but also the valuation which assigns the value T to all atomic formulas and the value F to all non-atomic formulas. I will thus be concerned with the question of the extent to which V(L, R) contains non-standard valuations, where L is a system of propositional calculus with the connectives $\wedge$, $\vee$, $\sim$, and $\supset$, and where R is the system of rules of inference given in [3][2]. The non-standard valuation just described is in fact excluded from V(L, R), since R allows $\sim\sim A$ to be inferred from A, for any A; if p is atomic, then the given valuation will make p T and $\sim\sim p$ F, but then a false conclusion can be inferred by R from a true premise, whence the valuation in question does not belong to V(L, R).

I will now exhibit Thomason's rules of inference in an informal graphic form. The vertical lines indicate subproofs, and the horizontal lines separate 'suppositions' from formulas that are inferred from earlier lines. Any formula may occur as a

'supposition' in any subproof. The hierarchic structure indicated in the diagrams is significant but the order of the premises is not. The lines are not necessarily consecutive.



There is an additional rule that is not readily expressible in the above graphic form: at any point in a proof, any line of a superordinate proof may be repeated.

The above rules of inference force ∧ to be truth-functional and indeed to have the standard truth-table. The first line of the table (T∧T: T) follows from ∧-introduction: if both A and B are true, then A∧B must be also, since if it were not, it would be a false conclusion derivable from true premises by the given rules. The other three lines of the table follow from ∧-elimination: if either A or B is false, then A∧B must be false also, since if it were true, a false conclusion (A or B as the case may be) would be deducible from the true premise A∧B.

∧ is in fact the only one of the four connectives whose truth-functionality is forced by the given rules. The rules force the first three lines of the truth-tables of v and ⊃, but not the fourth line (FvF: F; F⊃F: T). The first three lines of the table for v follow from v-introduction: if either A or B is true, then AvB must be also, since if it were false it would be a false conclusion derivable from a true premise (A or B, as the case may be). The second line of the truth table for ⊃ follows from ⊃-elimination: if A is true and B false, then A⊃B must be false, since if it were true, a false conclusion (B) would be derivable from true premises (A, A⊃B). The first and third lines of the standard truth table follow from the fact that (B⊃(A⊃B)) is a theorem in the given system and that a theorem, since deducible from any premises whatever, is deducible from true premises and thus itself true in any valuation in V(L, R)[3]. Thus, if B is true, then A⊃B must also be true, since otherwise a false conclusion (A⊃B) would be deducible from true premises (B, (B⊃(A⊃B))); thus the first line (T⊃T: T) and the third line (F⊃T: T) of the standard truth table are forced by the given rules of inference. However, the fourth lines of the standard tables for v and ⊃ do not follow, as I will prove shortly by showing that V(L, R) contains a

valuation in which some instances of F v F are true and others false, and some instances of F ⊃ F are true and others false.

First, however, let us turn to negation. I will first rule out of consideration one valuation which poses a trivial obstacle to the truth-functionality of negation. As things stand, V(L, R) contains the absurd valuation in which all propositions are assigned the value T. Indeed, no matter what rules R contained, V(L, R) would contain that valuation, since if all propositions are true in a given valuation, then no matter what the rules of inference are, they lead from true premises to true conclusions. Let me then restrict the notion of 'valuation' so as to include only those functions which assign the value F to at least one proposition[4]. Under this extra assumption, which will be taken for granted from here on, the first line of the truth table for ~ is forced by the rules. The given rules allow any proposition whatever to be inferred from contradictory premises; hence if A and ~A were both assigned T by some valuation, there would be a false proposition that was derivable from them (since in every valuation there are false propositions, and any proposition is inferrable from any pair of contradictory propositions), which means that the valuation is not in V(L, R). Thus the rules of inference force the first line of the standard table: when A is T, ~A must be F. However, the other line of the standard table for ~ is not forced, and the valuation which shows the fourth lines of the tables of v and ⊃ not to be forced will show the same for the second line of the table of ~: there will be both instances of (~F: T) and of (~F: F).

The valuation is as follows[5]: assign to a formula the value 'T' if it is a theorem in the given system and the value 'F' if it is not a theorem. This valuation belongs to V(L, R), as can be shown by considering each of the rules of inference in turn and drawing the trivial conclusion that it will derive therems from theorems. With respect to this valuation, there are instances of F v F that are true and other instances that are false. For example, if p is an atomic proposition, then p, ~p, and p v p are non-theorems and are thus assigned the value F, whereas p v ~p is a theorem and is thus assigned the value T. Hence p v p is an instance of (F v F: F), and p v ~p is an instance of (F v F: T). If p and q are atomic propositions, then p, q, and p ⊃ q are assigned the value F, whereas p ⊃ p is assigned the value T. Thus p ⊃ q is an instance of (F ⊃ F: F), and p ⊃ p is an instance of (F ⊃ F: T). Finally, if p is an atomic proposition, then both it and its negation are assigned the value F, whereas ~(p ⊃ p) is assigned the value F and ~~(p ⊃ p) the value T; thus ~p is an instance of (~F: F) and ~~(p ⊃ p) is an instance of (~F: T). Thus, none of v, ⊃, and ~ is truth-functional relative to V(L, R).

The following tables thus indicate the extent to which the given rules of inference force truth-functionality:

A	~A
T	F
F	T/(F)

A	B	A ∧ B	A v B	A ⊃ B
T	T	T	T	T
T	F	F	T	F
F	T	F	T	T
F	F	F	(T)/F	T/(F)

The / indicates that both true instances and false instances occur in V(L, R). Parentheses mark truth values that need not be attested in every valuation of V(L, R), whereas the unparenthesized truth values must be attested in every valuation. Since A ⊃ A is a theorem, there are instances of (F ⊃ F: T) in every valuation. Since ~(A ⊃ A) is a formula whose negation is a theorem, there are instances of (~F: T) in every valuation. And since ~(A ⊃ A) v ~(A ⊃ A) is also a formula whose negation is a theorem, there are also instances of (F v F: F) in every valuation.

For any valuation, truth-functionality in any of the three T/F cells implies truth-functionality in the other two. (1) Suppose that in a particular valuation in V(L, R), the negations of false propositions are all true. If A and B are both false, then ~A is true, ~B is true, ~A∧~B is true, and since A∨B is deductively equivalent to ~(~A∧~B), A∨B will be false. (2) Suppose that in some valuation in V(L, R), all disjunctions of false propositions are false. Then, since A∨~A is a theorem and thus true, A and ~A cannot both be false, since if they were they would be false propositions whose disjunction was true, contrary to the assumption. (3) Suppose that in some valuation in V(L, R), all instances of F⊃F are T. Let A be true and B false. If ~B were false, then ~B⊃~A would be an instance of F⊃F and thus true, and A⊃B, which is deducible from it, would also be true. But since A⊃B is an instance of T⊃F and is thus false, ~B couldn't have been false, and thus the negation of any false proposition is true. (4) Finally, suppose that in some valuation in V(L, R), the negation of every false proposition is true. Suppose that A, B, and A⊃B are all false. Then ~(A⊃B), being the negation of a false proposition, is true, and A∧~B, which is inferrable from it, is also true. But then A, which is inferrable from that, is true, contradicting the assumption that it was false. Thus, when A and B are false, A⊃B must be true.

Putting these results together, we obtain the equivalence of the following for any subset V of V(L, R):

(a) with respect to V, ~ is truth-functional.
(b) with respect to V, ∨ is truth-functional.
(c) with respect to V. ⊃ is truth-functional.
(d) V consists only of standard valuations.

All non-standard valuations in V(L, R) thus involve cases where a proposition and its negation are simultaneously false. The idea of something and its negation being simultaneously false is not quite as outlandish as it might seem at first. Suppose that propositions having a false semantic presupposition (e.g., the proposition that Richard Nixon regrets being the most popular President in the history of the USA) are considered not to lack a truth value (as in most previous treatments, e.g., [1], [4]), but that instead they and their negations are said to be false. Treating A and ~A as simultaneously false would amount to taking a broader conception of falsehood than in the approach which takes them to lack a truth value: in the broad conception of falsehood, A is false when A is not true, whereas in the narrow conception of falsehood, A is false when ~A is true. The two conceptions differ precisely in the case where neither A nor ~A is true (e.g., the case of 'Nixon regrets being the most popular President in the history of the USA'): that case is included in the broad conception of falsehood but excluded from the narrow conception.

There is a natural correspondence between the truth value assignments that are based on the narrow conception of falsehood and those based on the broad conception, namely:

narrow	
A	~A
t	f
f	t
#	#

broad	
A	~A
T	F
F	T
F	F

Here 'F' designates the broad conception of falsehood, 'f' the narrow conception of falsehood, and '#' designates lack of truth value; 't' means the same as 'T' but is kept typographically distinct so as to facilitate discussions in which the two treatments

are contrasted. I will speak of # as a 'third truth value', though I do not see any substance to the question of whether it is 'a truth value'; the fact that it is something that can appropriately fill a cell in a truth table is sufficient to make me happy to speak of it as a truth value.

An interesting question can now be raised: If the system that allows non-standard valuations is reinterpreted in terms of a narrow conception of falsehood, how do the resulting truth tables compare with the truth tables that follow from treatments of presupposition in terms of truth-value gaps, e.g., that of van Fraassen [4]? To illustrate the translation of T/F truth tables into t/f # truth tables, let us look at the case of ∧. We will be constructing a truth table containing 9 cells, representable in the following matrix form:

A \ B	t	f	#
t			
f			
#			

The upper left cell must contain t, since T∧T is always T, and T = t. The square to its right must have f, since if A is T, B F, and ~B T, then A∧B is F, and ~(A∧B), being deductively equivalent to (~A) v (~B), will be T, thus making A∧B be f. The upper right cell will have #. If A is T, B F, and ~B F, then A∧B is F, and ~(A∧B) will have the same truth value as ~A v ~B, which could conceivably be either T or F; however, in this case it could not be T; if ~A v ~B were T, then since A is T, A∧(~A v ~B) would be T, and, since distribution of ∧ over v and cancellation of contradictory disjuncts are valid derived rules of inference in this system, A∧~B will also be T, which contradicts the assumption that ~B is F. Thus both A∧B and its negation are F in the two-valued system, which means that A∧B is # in the three-valued system. Continuing in this fashion, the following table emerges:

A \ B	t	f	#
t	t	f	#
f	f	f	f
#	#	f	f/#

Two values are possible in the case of #∧#, since there are some choices of A # and B # for which ~A v ~B is T and others for which it is F; for example, in the valuation which assigns T to theorems and F to non-theorems, p∧q will be an instance of #∧#: #, and p∧~p will be an instance of #∧#: f. The tables for the other three connectives are as follows:

A	~A
t	f
f	t
#	#

A \ B	t	f	#
t	t	t	t
f	t	f	#
#	t	#	t/f/#

A v B

A \ B	t	f	#
t	t	f	f
f	t	t	t
#	t	#	t/#

A ⊃ B

These truth tables are identical to the truth tables that follow from supervaluations as defined by van Fraassen [4]. The supervaluation v_X defined by a set X of formulas is as follows:

If $v(A) = T$ for all classical valuations v that satisfy X, then $v_X(A) = t$.
If $v(A) = F$ for all classical valuations v that satisfy X, then $v_X(A) = f$.
Otherwise, $v_X(A) = \#$.

For any particular set X, these conditions impose sharp limitations on the relationship between the value that v_X assigns to a complex formula and the values that it assigns to its constituents, and a tabulation of the possible relationships in fact coincides with the truth tables just given. It is not surprising that the two approaches should yield the same truth tables, since the only fundamental difference between them is whether they are based on provability or on semantic validity. The valuations that comprise V(L, R) can be identified as those induced by sets of formulas in the following manner: for any set X of formulas, the (not necessarily standard) valuation w_X induced by X is given by the conditions

If $X \vdash A$, then $w_X(A) = t$.
If $X \vdash \sim A$, then $w_X(A) = f$.
Otherwise, $w_X(A) = \#$.

The only difference between this definition of w_X and van Fraassen's definition of v_X is that $\vdash$ appears in the former where 'semantically entail' appears in van Fraassen's definition. Since a semantic completeness theorem holds for the system of propositional logic which has the given rules of inference and the classical standard valuations, $v_X = w_X$ for every set of formulas X.

Thus, while the given set of rules of inference does not force all valuations to be 'standard', it is intimately connected with 'standard valuations' in that the set of valuations that conform to it are identical to the supervaluations defined in terms of standard classical valuations. An interesting topic for future research would be the effect on the class of valuations V(L, R) if a weaker system R of rules of inference were substituted, for example, if the given system were replaced by one which lacked the unnamed ninth rule ('at any point in a subproof, any line of a superordinate proof may be repeated') but had slightly generalized versions of the 'elimination' rules which allowed the premise containing the connective that is 'eliminated' to be superordinate to rather than just in the same subproof as the conclusion. This would be a weakening of the rules of inference, since it would not have the effect of adding any new proofs but would rule out proofs in which ~-elimination involved repetition of a superordinate line, e.g., it would rule out the best-known proof that from a contradiction anything follows. I have not established whether an alternative proof of that result is possible in the weakened system.

FOOTNOTES

[1] My term 'standard' should be kept distinct from van Fraassen's term 'classical'. A non-classical valuation is one in which there are truth-value gaps, that is, a valuation which fails to assign any truth value to some propositions, though the standard truth tables hold when the elementary propositions have truth values. A non-standard valuation is a valuation in which all propositions have truth values but the standard truth tables are violated.

[2] I include four connectives rather than just two, rather than following the usual policy of 'defining' two of the binary connectives in terms of ~ and the remaining binary connective, since it cannot be taken for granted that the rules of inference for two connectives would impose the same degree of truth-functionality as would the rules for all four connectives together.

[3] But cf. footnote 4.

[4] Whether an additional assumption is needed to exclude from V(L, R) the valuation which is identically F depends on a technicality: whether a proposition that is inferred from an empty set of premises (or is inferred from premises that are all irrelevant to it) shall count as 'inferred from true premises'. I adopt here the interpretation under which theorems count as 'inferred from true premises'.

[5] I am grateful to Richmond Thomason for providing me with this proof. Thomason notes that any of a more general class of valuations would do as well: if X is a set of formulas such that there is at least one formula A for which neither $X \vdash A$ nor $X \vdash \sim A$, then the valuation defined by '$w_X(A) = T$ if $X \vdash A$; $w_X(A) = F$ otherwise' would also render $\vee$, $\supset$, and $\sim$ non-truth-functional. The results presented below imply that these are the only non-standard valuations in V(L, R).

REFERENCES

[1] L. Karttunen, "Presuppositions of compound sentences", Linguistic Inquiry, vol. 4, pp. 169-193, 1973.

[2] A. N. Prior, "The runabout inference ticket", Analysis, vol. 21, pp. 38-39, 1960; also appears in P. F. Strawson, Philosophical Logic, Oxford University Press, 1967.

[3] R. Thomason, Symbolic logic: an introduction, Macmillan, 1970.

[4] B. van Fraassen, "Presuppositions, supervaluations, and free logic", in K. Lambert, The logical way of doing things, Yale University Press, 1969, pp. 67-91.

ON EQUIVALENTIAL ALGEBRAS

Jacek K. Kabziński
Jagiellonian University
Poland

Andrzej Wroński
Jagiellonian University
Poland

A notable study of the equivalential fragment of the intuitionistic propositional logic (INT) by means of proof-theoretical methods was given by R.E. Tax [8] to whom belongs the credit of finding the first axiomatization. Our task is to give a characterization of this fragment from an algebraic point of view. We introduce the notion of equivalential algebra being an algebraic counterpart of the equivalential fragment of INT and prove some theorems characterizing it. The result giving a relatively good insight into the structure of equivalential algebras is the embedding theorem by which every equivalential algebra can be embedded (in a reasonable sense) into a meet-semilattice with relative pseudocomplementation.

Throughout this paper we use a common algebraic notation and nomenclature taken mainly from [4] and [5]. In particular, the German capitals, early in the alphabet $\mathfrak{A},\mathfrak{B},\ldots$ will be used for algebras and the corresponding Latin capitals $A,B,\ldots$ for their domains. The letters late in the Latin capitals alphabet X,Y,Z will be reserved for finite sets of elements of an algebra and the corresponding bold-face capitals $\mathbf{X},\mathbf{Y},\mathbf{Z}$ for finite families of such sets. We occasionally denote a finite set by listing its members in braces. For example $\{X\}$ denotes the singleton containing X as the only element.

By an equivalential algebra it is meant an algebra $\mathfrak{A} = \langle A, \leftrightarrow \rangle$ with a binary operation $\leftrightarrow$ (called the equivalence operation) satisfying the following postulates for arbitrary $a,b,c \in A$:

(E1) $\quad (a \leftrightarrow a) \leftrightarrow b = b$,
(E2) $\quad ((a \leftrightarrow b) \leftrightarrow c) \leftrightarrow c = (a \leftrightarrow c) \leftrightarrow (b \leftrightarrow c)$,
(E3) $\quad ((a \leftrightarrow b) \leftrightarrow ((a \leftrightarrow c) \leftrightarrow c)) \leftrightarrow ((a \leftrightarrow c) \leftrightarrow c) = a \leftrightarrow b$.

The class of all equivalential algebras will be denoted by EQ. By virtue of the definition, EQ is an equational class and therefore it is closed under formation of subalgebras, homomorphic images and direct products. Since we shall make frequent use of a number of equations derived from the postulates (E1),(E2) and (E3) then for simplifying the notations we adopt the convention of associating to the left and ignoring the equivalence sign $\leftrightarrow$. For example (E1),(E2) and (E3) should be abbreviated as follows:

(E1) $\quad aab = b$,
(E2) $\quad abcc = ac(bc)$,
(E3) $\quad ab(acc)(acc) = ab$.

The following lemma provides a list of derived equations which will be needed in the sequel:

LEMMA 1. If $\mathfrak{A} \in EQ$, $a,b,c \in A$ then the following equations hold:

(i) $ab = ba$,
(ii) $aa = bb$,
(iii) $abb(baa) = ab$,
(iv) $abbb = ab$,
(v) $abcc = acc(bcc)$,
(vi) $abbaa = abb$,
(vii) $abbcc = accbb$,
(viii) $abb(bcc)(bcc) = abb$,
(ix) $abb(bc)(bc) = abbcc$,
(x) $a(bc)b = abcb$.

PROOF. (i) $ab =$(E1) $bb(ab) =$(E2) $babb =$(E1) $ba(bbb)(bbb) =$(E3) ba .

(ii) $aa =$(E1) $bb(aa) =$(i) $aa(bb) =$(E1) bb .

(iii) $abb(baa) =$(i) $a(ab)(b(ab)) =$(E2) $ab(ab)(ab) =$(E1) ab .

(iv) $abbb =$(i) $babb =$(E2) $bb(ab) =$(E1) ab .

(v) $abcc =$(iv) $abcccc =$(E2) $ac(bc)cc =$(E2) $acc(bcc)$.

(vi) $abbaa =$(i) $b(ba)aa =$(E2) $ba(baa) =$(i) $ab(baa) =$(iii)
$abb(baa)(baa) =$(i) $b(ab)(baa)(baa) =$(E3) $b(ab) =$(i) abb .

(vii) $abbcc =$(v) $acc(bcc)(bcc) =$(v)
$a(bcc)(bcc)(c(bcc)(bcc))(c(bcc)(bcc)) =$(i)
$a(bcc)(bcc)(bccc(bcc))(bccc(bcc)) =$(iv) $a(bcc)(bcc)(bc(bcc))(bc(bcc))$
$=$(iii) $a(bcc)(bcc)(bcc(cbb)(bcc))(bcc(cbb)(bcc)) =$(i)
$a(bcc)(bcc)(b(cb)(bcc)(bcc))(b(cb)(bcc)(bcc)) =$(E3)
$a(bcc)(bcc)(b(cb))(b(cb)) =$(i) $a(bcc)(bcc)(cbb)(cbb) =$(v)
$a(cbb)(cbb)(bcc(cbb)(cbb))(bcc(cbb)(cbb)) =$(iii)
$a(cbb)(cbb)(bc(cbb))(bc(cbb)) =$(i) $a(cbb)(cbb)(cb(cbb))(cb(cbb)) =$(iv)
$a(cbb)(cbb)(cbbb(cbb))(cbbb(cbb)) =$(i)
$a(cbb)(cbb)(b(cbb)(cbb))(b(cbb)(cbb)) =$(v) $abb(cbb)(cbb) =$(v) $accbb$.

(viii) $abb(bcc)(bcc) =$(i) $b(ab)(bcc)(bcc) =$(E3) $b(ab) =$(i) abb .

(ix) $abb(bc)(bc) =$(E2) $ab(bc)(b(bc)) =$(i) $ab(cb)(cbb) =$(E2) $acbb(cbb)$
$=$(v) $accbb =$(vii) $abbcc$.

(x) $a(bc)b =$(iv) $a(bc)bbb =$(v) $abb(bcbb)b =$(i) $abb(cbbb)b =$(iv)
$abb(cb)b =$(E2) $abcbbb =$(iv) $abcb$. Q.E.D.

By LEMMA 1(i),(ii) it follows that every equivalential algebra $\mathfrak{A}$ has an (unique) unit element $\mathbf{1}$ such that for every $a \in A$, $\mathbf{1}a = a\mathbf{1} = a$ (putting $\mathbf{1} = cc$ for some $c \in A$ we get the element with required properties). Moreover, for every $a,b \in A$, $ab = \mathbf{1}$ iff $a = b$. Indeed, if $ab = \mathbf{1}$ then $a =$(E1) $cca = \mathbf{1}a = aba =$(i) $baa =$(vi) $baabb$
$=$(i) $ababb = \mathbf{1}abb = ccabb =$(E1) $abb = \mathbf{1}b = ccb =$(E1) b .

By a filter of an equivalential algebra $\mathfrak{A}$ we mean a non-empty subset $\Phi \subseteq A$ such that for every $a,b \in A$:

(i) if $a \in \Phi$ then $abb \in \Phi$,
(ii) if $a, ba \in \Phi$ then $b \in \Phi$.

Let $F(\mathfrak{A})$ be the set of all filters of $\mathfrak{A}$. It is easy to check that any intersection of a family of filters is not empty (for every $\Phi \in F(\mathfrak{A})$, $1 \in \Phi$) and therefore $F(\mathfrak{A})$ is closed under arbitrary intersections. This yields that $F(\mathfrak{A})$ forms a complete lattice with respect to the set-theoretical inclusion. To characterize this lattice let us note the following:

THEOREM 1. (i) If $\mathfrak{A} \in EQ$ and $\Phi \in F(\mathfrak{A})$ then the relation $\nabla_\Phi = \{\langle a,b\rangle : ab \in \Phi\}$ is a congruence of $\mathfrak{A}$ and $[1]\nabla_\Phi = \Phi$;

(ii) If $\mathfrak{A} \in EQ$ and Θ is a congruence of $\mathfrak{A}$ then $[1]\Theta \in F(\mathfrak{A})$ and $\nabla_{[1]\Theta} = \Theta$.

PROOF. This fact follows by an easy computation based on LEMMA 1 and the properties of the unit element of $\mathfrak{A}$. Q.E.D.

COROLLARY. For every $\mathfrak{A} \in EQ$ the mapping $\Phi \mapsto \nabla_\Phi$ sets up an isomorphism between the lattice $\langle F(\mathfrak{A}),\subseteq\rangle$ and the lattice of all congruence relations of $\mathfrak{A}$.

If $\mathfrak{A} \in EQ$ and $M \subseteq A$ then by $[M)$ we denote the smallest filter of $\mathfrak{A}$ containing M . If M is a singleton, say $M = \{a\}$ then we shall write $[a)$ instead of $[M)$. In view of THEOREM 1 for every $M \subseteq A$ we have the corresponding congruence relation $\nabla_{[M)}$ such that $a \equiv b(\nabla_{[M)})$ iff $ab \in [M)$. For brevity we shall write in the sequel: $\mathfrak{A}/M$, A/M , $[a]M$, $a \equiv_M b$ instead of: $\mathfrak{A}/\nabla_{[M)}$, $A/\nabla_{[M)}$, $[a]\nabla_{[M)}$, $a \equiv b(\nabla_{[M)})$ respectively.

For every $\mathfrak{A} \in EQ$ we define a family of mappings $\{\&X : X \subseteq A$, $|X| < \aleph_0\}$ putting for every $a,b \in A$, $a\&\emptyset = a$ and $a\&X\cup\{b\} = (a\&X)bb$. For example, if $X = \{b_1,\ldots,b_n\}$, $a \in A$ then $a\&X = ab_1b_1\ldots b_nb_n$. It follows from LEMMA 1(iv),(vii) that all the mappings $\&X$ are well defined since the value of $\&X$ does not depend on permuting or repeating some elements of X .

LEMMA 2. If $\mathfrak{A} \in EQ$, $M \subseteq A$, $a,b \in A$ and X,Y are finite subsets of A then the following conditions are satisfied:

(i) $abb = a\&\{b\}$,
(ii) if $a \equiv_M b$ then $a\&X \equiv_M b\&X$,
(iii) $1\&X = 1$,
(iv) $a\&X \in [a)$,
(v) $(ab)\&X = (a\&X)(b\&X)$,
(vi) $(a\&X)\&Y = a\&X\cup Y$,
(vii) $(a\&X)\&Y = (a\&Y)\&X$,
(viii) $(a\&X)\&X = a\&X$,
(ix) if $b \in [X)$ then $(abb)\&X = a\&X$,
(x) if $Y \subseteq [X)$ then $(a\&Y)\&X = a\&X$.

PROOF. To verify (i) observe that $abb = (a\&\emptyset)bb = a\&\emptyset\cup\{b\} = a\&\{b\}$. The conditions (ii),...,(vi) follow by a routine induction argument with respect to the number of elements of X. The conditions (vii) and (viii) can be obtained directly from (vi).

Proving (ix) we make use of (i),(vi),(vii) and LEMMA 1(i), (viii),(ix). Suppose first that $b \in X$. Then $(abb)\&X = (a\&\{b\})\&X = a\&\{b\}\cup X = a\&X$. Suppose now that $(abb)\&X = a\&X$. Then $(a(bcc)(bcc))\&X = (a\&\{bcc\})\&X = (a\&X)\&\{bcc\} = ((abb)\&X)\&\{bcc\} = ((abb)\&\{bcc\})\&X = (abb(bcc)(bcc))\&X = (abb)\&X = a\&X$. Suppose finally that $(abb)\&X = a\&X$ and $(a(cb)(cb))\&X = a\&X$. Then $(acc)\&X = (a\&\{c\})\&X = (a\&X)\&\{c\} = ((abb)\&X)\&\{c\} = ((abb)\&\{c\})\&X = (abbcc)\&X = (abb(bc)(bc))\&X = (abb(cb)(cb))\&X = ((abb)\&\{cb\})\&X = ((abb)\&X)\&\{cb\} = (a\&X)\&\{cb\} = (a\&\{cb\})\&X = (a(cb)(cb))\&X = a\&X$.

Proving (x) we shall use an induction argument with respect to the number of elements of Y. Disregarding the trivial case $Y = \emptyset$ we assume that (x) holds for a finite $Y \subseteq A$ and $Y\cup\{b\} \subseteq [A)$. We need to show that (x) holds for $Y\cup\{b\}$. Indeed, using (i),(vi),(vii) and (ix) we get $(a\&Y\cup\{b\})\&X = ((a\&Y)\&\{b\})\&X = ((a\&Y)\&X)\&\{b\} = (a\&X)\&\{b\} = (a\&\{b\})\&X = (abb)\&X = a\&X$ which completes the proof. Q.E.D.

COROLLARY. If $\mathfrak{A} \in EQ$ and X is a finite subset of A then the mapping $\&X$ is a retraction of $\mathfrak{A}$.

For every $\mathfrak{A} \in EQ$ and $a \in A$ we define an algebra $\mathfrak{A}_a = \langle A_a, \leftrightarrow\rangle$ as the subalgebra of $\mathfrak{A}$ generated by the set $\{a\&X : X \subseteq A, |X| < \aleph_0\}$.

THEOREM 2. If $\mathfrak{A} \in EQ$, $a \in A$ then the algebra $\mathfrak{A}_a$ is associative.

PROOF. By induction one can show that $A_a \subseteq [a)$ and for every $b \in A_a$, $baa = b$. Next, if $b,c,d \in A_a$ then both $b(cd),bcd$ belong to A_a and therefore applying LEMMA 1(x), LEMMA 2(i),(ix) we obtain $bcd = bcdaa = bcdccaa = b(cd)ccaa = b(cd)aa = b(cd)$ which was to be proved. Q.E.D.

For every $\mathfrak{A} \in EQ$ we define a family of mappings $\{*\mathbf{X} : \mathbf{X}$ is a finite set of finite subsets of A$\}$ putting for every $a \in A$ and every finite $X \subseteq A$, $a*\emptyset = 1$, $a*\mathbf{X}\cup\{X\} = (a*\mathbf{X}-\{X\})(a\&X)$. For example, if $\mathbf{X} = \{X_1,...,X_n\}$ then $a*\mathbf{X} = (a\&X_1)...(a\&X_n)$. By LEMMA 1(i), THEOREM 2 the value of $*\mathbf{X}$ does not depend on permuting or repeating of elements of $\mathbf{X}$.

LEMMA 3. If $\mathfrak{A} \in EQ$, $M \subseteq A$, $a,b \in A$, X is a finite subset of A and $\mathbf{X},\mathbf{Y},\mathbf{Z}$ are finite sets of finite subsets of A then the following conditions hold:

(i) $a\&X = a*\{X\}$,
(ii) if $a \equiv_M b$ then $a*\mathbf{X} \equiv_M b*\mathbf{X}$,
(iii) $a*\mathbf{X} \in A_a$,
(iv) $(a*\mathbf{X})\&Y \in A_a$,
(v) $(a*\mathbf{X})*\mathbf{Y} \in A_a$,
(vi) $(a*\mathbf{X})(a*\mathbf{Y}) = a*\mathbf{X}\div\mathbf{Y}$ ($\div$ is the symmetric difference of sets),
(vii) $((a*\mathbf{X})(a*\mathbf{Y}))*\mathbf{Z} = ((a*\mathbf{X})*\mathbf{Z})((a*\mathbf{Y})*\mathbf{Z})$,
(viii) $(a*\mathbf{X})\&Y = (a\&Y)*\mathbf{X}$,
(ix) $(a*\mathbf{X})*\mathbf{Y} = (a*\mathbf{Y})*\mathbf{X}$,
(x) $(a*\mathbf{X})*\mathbf{X} = a*\mathbf{X}$.

PROOF. The conditions (i),...,(iv) are obvious and hardly need a proof. To prove (v) observe first that $(a*\mathbf{X})*\emptyset = \mathbf{1} \in A_a$ which means that (v) holds if $\mathbf{Y} = \emptyset$. Suppose that (v) holds for a set $\mathbf{Y}$ and $Y \notin \mathbf{Y}$. We need to show that (v) holds also for $\mathbf{Y} \cup \{Y\}$. Indeed, using (iv) we get $(a*\mathbf{X})*\mathbf{Y}\cup\{Y\} = ((a*\mathbf{X})*\mathbf{Y})((a*\mathbf{X})\&Y) \in A_a$.

To prove (vi) let us note that $(a*\mathbf{X})(a*\emptyset) = (a*\mathbf{X})\mathbf{1} = a*\mathbf{X}\div\emptyset$ which means that (vi) holds in the case $\mathbf{Y} = \emptyset$. Suppose now that (vi) holds for $\mathbf{Y}$ and $Y \notin \mathbf{Y}$. To show that (vi) holds for $\mathbf{Y}\cup\{Y\}$ let us consider first the case $Y \in \mathbf{X}$. Then $Y \in \mathbf{X}\div\mathbf{Y}$, $(\mathbf{X}\div\mathbf{Y})\cup\{Y\} = \mathbf{X}\div\mathbf{Y}$, $(\mathbf{X}\div\mathbf{Y})-\{Y\} = \mathbf{X}\div(\mathbf{Y}\cup\{Y\})$ and therefore $(a*\mathbf{X})(a*\mathbf{Y}\cup\{Y\}) = (a*\mathbf{X})((a*\mathbf{Y})(a\&Y)) = (a*\mathbf{X})(a*\mathbf{Y})(a\&Y) = (a*\mathbf{X}\div\mathbf{Y})(a\&Y) = (a*(\mathbf{X}\div\mathbf{Y})\cup\{Y\})(a\&Y) = (a*(\mathbf{X}\div\mathbf{Y})-\{Y\})(a\&Y)(a\&Y) = (a*(\mathbf{X}\div\mathbf{Y})-\{Y\})((a\&Y)(a\&Y)) = a*(\mathbf{X}\div\mathbf{Y})-\{Y\} = a*\mathbf{X}\div(\mathbf{Y}\cup\{Y\})$. Let us consider finally the case $Y \notin \mathbf{X}$. Then $Y \notin \mathbf{X}\div\mathbf{Y}$, $(\mathbf{X}\div\mathbf{Y})\cup\{Y\} = \mathbf{X}\div(\mathbf{Y}\cup\{Y\})$ and therefore $(a*\mathbf{X})(a*\mathbf{Y}\cup\{Y\}) = (a*\mathbf{X})((a*\mathbf{Y})(a\&Y) = (a*\mathbf{X})(a*\mathbf{Y})(a\&Y) = (a*\mathbf{X}\div\mathbf{Y})(a\&Y) = a*(\mathbf{X}\div\mathbf{Y})\cup\{Y\} = a*\mathbf{X}\div(\mathbf{Y}\cup\{Y\})$.

To prove (vii) observe that it holds in the case $\mathbf{Z} = \emptyset$ because $((a*\mathbf{X})(a*\mathbf{Y}))*\emptyset = \mathbf{1} = \mathbf{1}\mathbf{1} = ((a*\mathbf{X})*\emptyset)((a*\mathbf{Y})*\emptyset)$. If (vii) holds for $\mathbf{Z}$ and $Z \notin \mathbf{Z}$ then $((a*\mathbf{X})(a*\mathbf{Y}))*\mathbf{Z}\cup\{Z\} = (((a*\mathbf{X})(a*\mathbf{Y}))*\mathbf{Z})(((a*\mathbf{X})(a*\mathbf{Y}))\&Z) = (((a*\mathbf{X})*\mathbf{Z})((a*\mathbf{Y})*\mathbf{Z}))(((a*\mathbf{X})\&Z)((a*\mathbf{Y})\&Z)) = (((a*\mathbf{X})*\mathbf{Z})((a*\mathbf{X})\&Z))(((a*\mathbf{Y})*\mathbf{Z})((a*\mathbf{Y})\&Z)) = ((a*\mathbf{X})*\mathbf{Z}\cup\{Z\})((a*\mathbf{Y})*\mathbf{Z}\cup\{Z\})$.

To prove (viii) observe that $(a*\emptyset)\&Y = \mathbf{1}\&Y = \mathbf{1} = (a\&Y)*\emptyset$ which means that it holds in the case $\mathbf{X} = \emptyset$. Suppose that (viii) holds for $\mathbf{X}$ and $X \notin \mathbf{X}$. Then $(a*\mathbf{X}\cup\{X\})\&Y = ((a*\mathbf{X})(a\&X))\&Y = ((a*\mathbf{X})\&Y)((a\&X)\&Y) = ((a\&Y)*\mathbf{X})((a\&Y)\&X) = (a\&Y)*\mathbf{X}\cup\{X\}$.

To prove (ix) note that it holds in the case $\mathbf{Y}=\emptyset$ because $(a*\mathbf{X})*\emptyset = \mathbf{1} = \mathbf{1}*\mathbf{X} = (\mathbf{1}*\emptyset)*\mathbf{X}$. If (ix) holds for $\mathbf{Y}$ and $Y \notin \mathbf{Y}$ then using (i),(vii),(viii) we get $(a*\mathbf{X})*\mathbf{Y}\cup\{Y\} = ((a*\mathbf{X})*\mathbf{Y})((a*\mathbf{X})\&Y) = ((a*\mathbf{Y})*\mathbf{X})((a\&Y)*\mathbf{X}) = ((a*\mathbf{Y})(a\&Y))*\mathbf{X} = (a*\mathbf{Y}\cup\{Y\})*\mathbf{X}$.

To prove (x) observe that $(a*\emptyset)*\emptyset = \mathbf{1} = a*\emptyset$ which proves (x) in the case $\mathbf{X} = \emptyset$. If (x) holds for $\mathbf{X}$ and $X \notin \mathbf{X}$ then by (i),(iii),(iv), (vii),(viii) we get $(a*\mathbf{X}\cup\{X\})*\mathbf{X}\cup\{X\} = (((a*\mathbf{X})(a\&X))*\mathbf{X})(((a*\mathbf{X})(a\&X))\&X) = (((a*\mathbf{X})*\mathbf{X})((a\&X)*\mathbf{X}))(((a*\mathbf{X})\&X)((a\&X)\&X)) = ((a*\mathbf{X})((a\&X)*\mathbf{X}))(((a\&X)*\mathbf{X})(a\&X)) = ((a\&X)*\mathbf{X})((a\&X)*\mathbf{X})((a*\mathbf{X})(a\&X)) = (a*\mathbf{X})(a\&X) = a*\mathbf{X}\cup\{X\}$ which completes the proof of lemma. Q.E.D.

THEOREM 3. If $\mathfrak{A} \in EQ$, $a,b \in A$ then the conditions below are equivalent:
(i) $b \in A_a$,
(ii) $b = a*\mathbf{X}$ for some finite set $\mathbf{X}$ of finite subsets of A .

The fact that (ii) implies (i) follows by LEMMA 3(iii) and the converse implication by LEMMA 3(i),(vi). Using THEOREM 3 and LEMMA 3 (vii),(x) one obtains the following:

COROLLARY. If $\mathfrak{A} \in EQ$, $a \in A$ and $\mathbf{X}$ is a finite set of finite subsets of A then the mapping $*\mathbf{X}$ is a retraction of A_a .

LEMMA 4. If $\mathfrak{A} \in EQ$, $\mathbf{M} \subseteq A$, $a,b \in A$ then the following conditions hold:
(i) $a \in [M)$ iff $a(c_1\&X_1)\ldots(c_n\&X_n) = \mathbf{1}$ for some finite sets $X_1,\ldots,X_n \subseteq A$ and elements $c_1,\ldots,c_n \in \mathbf{M}$,
(ii) $a \in [M\cup\{b\})$ iff $abb \equiv_M b*\mathbf{X}$ for some finite set $\mathbf{X}$ of finite subsets of A .

PROOF. In both the cases (i),(ii) the necessity is obvious. To prove the sufficiency in the case (i) observe that if $a \in M$ then $a(a\&\emptyset) = aa = \mathbf{1}$. Next, if $a(c_1\&X_1)\ldots(c_n\&X_n) = \mathbf{1}$ then $abb(c_1\&X_1\cup\{b\})\ldots(c_n\&X_n\cup\{b\}) = (a\&\{b\})(c_1\&X_1\cup\{b\})\ldots(c_n\&X_n\cup\{b\}) = (a\&\{b\})((c_1\&X_1)\&\{b\})\ldots((c_n\&X_n)\&\{b\}) = (a(c_1\&X_1)\ldots(c_n\&X_n))\&\{b\} = \mathbf{1}\&\{b\} = \mathbf{1}$. Suppose that $a(c_1\&X_1)\ldots(c_n\&X_n) = \mathbf{1}$, $ba(d_1\&Y_1)\ldots(d_m\&Y_m) = \mathbf{1}$ and put $Z = \{c_1\&X_1,\ldots,c_n\&X_n\}$. Then we get $b(c_1\&X_1)\ldots(c_n\&X_n)(d_1\&Y_1\cup Z)\ldots(d_m\&Y_m\cup Z) = b(c_1\&X_1)\ldots(c_n\&X_n)\mathbf{1}(d_1\&Y_1\cup Z)\ldots(d_m\&Y_m\cup Z) = b(c_1\&X_1)\ldots(c_n\&X_n)(a(c_1\&X_1)\ldots(c_n\&X_n))(d_1\&Y_1\cup Z)\ldots(d_m\&Y_m\cup Z) = ((ba)\&Z)((d_1\&Y_1)\&Z)\ldots((d_m\&Y_m)\&Z) = ((ba)(d_1\&Y_1)\ldots(d_m\&Y_m))\&Z = \mathbf{1}\&Z = \mathbf{1}$.

To prove the sufficiency in the case (ii) observe that if $a \in M$ then $abb \equiv_M \mathbf{1} = b*\emptyset$. Next, if $a=b$ then $abb = bbb = b = b*\{\emptyset\}$. Suppose that $abb \equiv_M b*\mathbf{X}$. Then $accbb = abbcc \equiv_M (b*\mathbf{X})cc = (b*\mathbf{X})\&\{c\}$. Next by LEMMA 3(iv) and THEOREM 3 we know that there exists $\mathbf{Y}$ such that $(b*\mathbf{X})\&\{c\} = b*\mathbf{Y}$. This gives that $accbb \equiv_M b*\mathbf{Y}$. Suppose finally that $abb \equiv_M b*\mathbf{X}$ and $cabb \equiv_M b*\mathbf{Y}$. Then $cbb = cbb(b*\mathbf{X})(b*\mathbf{X}) \equiv_M cbb(abb)(b*\mathbf{X}) = cabb(b*\mathbf{X}) \equiv_M (b*\mathbf{Y})(b*\mathbf{X}) = b*\mathbf{Y}\div\mathbf{X}$ which finishes the proof. Q.E.D.

The following statement plays an important role in the theory of equivalential algebras:

THEOREM 4. If $\mathfrak{A} \in EQ$, $M \subseteq A$, $a,b \in A$ then the conditions below are equivalent:
(i) $a \equiv_M b$,
(ii) $[M\cup\{a\}) = [M\cup\{b\})$.

PROOF. It is obvious that (i) implies (ii). To show the converse we shall prove that $abb \equiv_M baa$ which gives $a \equiv_M b$ by LEMMA 1(iii). First observe that (ii) implies $a \in [M\cup\{b\})$ and $b \in [M\cup\{a\})$. Thus by LEMMA 4 (ii) we know that there exist finite sets $\mathbf{X},\mathbf{Y}$ of finite subsets of A such that $abb \equiv_M b*\mathbf{X}$ and $baa \equiv_M a*\mathbf{Y}$. Next, applying LEMMA 1(vii), LEMMA 2(i) and LEMMA 3(ii),(viii),(ix),(x) we get $abb = abbaa \equiv_M (b*\mathbf{X})aa = (baa)*\mathbf{X} = (baabb)*\mathbf{X} \equiv_M ((a*\mathbf{Y})bb)*\mathbf{X} = ((abb)*\mathbf{Y})*\mathbf{X} \equiv_M ((b*\mathbf{X})*\mathbf{Y})*\mathbf{X} = ((b*\mathbf{X})*\mathbf{X})*\mathbf{Y} = (b*\mathbf{X})*\mathbf{Y} \equiv_M (abb)*\mathbf{Y} = (a*\mathbf{Y})bb \equiv_M baabb = baa$. Q.E.D.

For every $\mathfrak{A} \in EQ$ and $M \subseteq A$ we define a binary relation $\leqslant_{\mathfrak{A}/M}$ in the quotient set A/M putting for every $a,b \in A$, $[a]M \leqslant_{\mathfrak{A}/M} [b]M$ iff $b \in [M\cup\{a\})$. By THEOREM 4 it follows immediately that the relation $\leqslant_{\mathfrak{A}/M}$ is a partial ordering in the quotient set A/M.

Since $\mathfrak{A}/\emptyset$ is isomorphic to $\mathfrak{A}$ then $\leqslant_{\mathfrak{A}/\emptyset}$ determines a partial ordering $\leqslant_{\mathfrak{A}}$ of the algebra $\mathfrak{A}$. Recall that $a \leqslant_{\mathfrak{A}} b$ iff $b \in [a)$. Obviously $\mathbf{1}$ is the greatest element in A with respect to $\leqslant_{\mathfrak{A}}$. We will say that an equivalential algebra $\mathfrak{A}$ is strongly compact (comp. [7]) iff there exists the greatest element in the set $A-\{\mathbf{1}\}$ with respect to $\leqslant_{\mathfrak{A}}$ (such an element of $\mathfrak{A}$ - if it exists - will be denoted by $\blacktriangle$).

It is known that every algebra is isomorphic to a subdirect product of subdirectly indecomposable algebras (see [1]). To characterize subdirectly indecomposable equivalential algebras let us note the following:

THEOREM 5. For every non-degenerate algebra $\mathfrak{A} \in EQ$ the following conditions are equivalent:
(i) $\mathfrak{A}$ is subdirectly indecomposable ,
(ii) $\mathfrak{A}$ is strongly compact .

PROOF. Recall that a non-degenerate algebra is subdirectly indecomposable iff it has the smallest non-trivial congruence relation. Thus in view of COROLLARY to THEOREM 1 the subdirect indecomposability of an equivalential algebra is equivalent to the existence of the smallest non-trivial filter. To show that (ii) implies (i) we shall prove that such a filter is $[\Delta)$. Indeed, if $\tilde{\Phi} \in F(\mathfrak{A})$ and $a \in \tilde{\Phi}-\{1\}$ then $a \leqslant_{\mathfrak{A}} \Delta$ which means that $\Delta \in [a)$ and thus $[\Delta) \subseteq [a) \subseteq \tilde{\Phi}$ which was to be shown. Conversely, to show that (i) implies (ii) let us suppose that $\tilde{\Phi}$ is the smallest non-trivial filter of $\mathfrak{A}$. Observe that each $a \in \tilde{\Phi}$ is an upper bound of $A-\{1\}$. Indeed, supposing contrary we get that for some $b \in A-\{1\}$ and $a \in \tilde{\Phi}$, $b \not\leqslant_{\mathfrak{A}} a$ which means that $a \notin [b)$ and therefore $[b)$ is non-trivial and $\tilde{\Phi} \not\subseteq [b)$ which is a contradiction. Since there exists a non-unit element in $\tilde{\Phi}$ then it is the greatest in $A-\{1\}$. Q.E.D.

Now we are in the position to discuss some equational subclasses of EQ. Let EL be the class of all equivalential algebras satisfying the following postulate:

(λ) $\quad (a(bcc)(bcc))(a(cbb)(cbb))(a(bc)(bc)) = a$.

To characterize strongly compact algebras in EL we need the following definition: $\mathfrak{A} \in EQ$ is linearly ordered iff $\langle A, \leqslant_{\mathfrak{A}} \rangle$ is a chain. Let us note the following:

LEMMA 5. For every $\mathfrak{A} \in EQ$ the following conditions are equivalent:
(i) $\quad \mathfrak{A}$ is linearly ordered ,
(ii) $\quad$ if $b,c \in A$, $b \neq c$ then either $bc = c$ or $cb = b$.

PROOF. The implication from (ii) to (i) is obvious. To prove the converse let us suppose that $\mathfrak{A}$ is linearly ordered, $b,c \in A$, $b \neq c$. Then $b \not\leqslant_{\mathfrak{A}} c$ or $c \not\leqslant_{\mathfrak{A}} b$. If $b \not\leqslant_{\mathfrak{A}} c$ then we get succesively: $c \notin [b)$, $bc \notin [b)$, $b \not\leqslant_{\mathfrak{A}} bc$. Since $\mathfrak{A}$ is linearly ordered then $c \leqslant_{\mathfrak{A}} b$ and $bc \leqslant_{\mathfrak{A}} b$ which gives $b \in [c)$ and $b \in [bc)$. Consequently $bc \in [c)$, $c \in [bc)$ which means that $c \leqslant_{\mathfrak{A}} bc$ and $bc \leqslant_{\mathfrak{A}} c$ proving the equality $bc = c$. In a very similar way the supposition $c \not\leqslant_{\mathfrak{A}} b$ yields the equality $cb = b$. Q.E.D.

THEOREM 6. If $\mathfrak{A} \subseteq EQ$ is strongly compact then the conditions below are equivalent :
(i) $\quad \mathfrak{A} \in EL$,
(ii) $\quad \mathfrak{A}$ is linearly ordered .

PROOF. The implication from (ii) to (i) follows directly by LEMMA 5. To prove the converse let us suppose that a strongly compact algebra $\mathfrak{A}$ is not linearly ordered i.e. $b \not\leqslant_{\mathfrak{A}} c$ and $c \not\leqslant_{\mathfrak{A}} b$ for some $b,c \in A$. We shall prove that $\Delta(bcc) = bcc$, $\Delta(cbb) = cbb$ and $\Delta(bc) = bc$ which imply that $\mathfrak{A} \notin EL$. Observe first that $bcc \neq 1$ and $bcc \neq \Delta$ (in the opposite case one obtains $bcc \in [c)$, $b \in [c)$ and finally $c \leqslant_{\mathfrak{A}} b$, a contradiction). Next we get succesively: $\Delta \in [bcc)$, $\Delta(bcc) \in [bcc)$, $bcc \leqslant_{\mathfrak{A}} \Delta(bcc)$ and also $\Delta \in [\Delta(bcc))$, $bcc \in [\Delta(bcc))$, $\Delta(bcc) \leqslant_{\mathfrak{A}} bcc$ which proves that $\Delta(bcc) = bcc$. The remaining equalities can be proved similarly. Q.E.D.

For every $\mathfrak{A} \in EQ$ and $n = 1,2,\ldots$ we define n-ary polynomial P_n of the algebra $\mathfrak{A}$ putting for every sequence $a_1,a_2,\ldots$ of elements of A , $P_1(a_1) = a_1$ and $P_{n+1}(a_1,\ldots,a_{n+1}) = a_{n+1}P_n(a_1,\ldots,a_n)P_n(a_1,\ldots,a_n)a_{n+1}$.

LEMMA 6. If $\mathfrak{A} \in EQ$ then for every $n,m = 1,2,\ldots$ and every sequence $a_1,\ldots,a_{n+m}$ of elements of A, $P_{n+m}(a_1,\ldots,a_{n+m})P_n(a_1,\ldots,a_n) = P_n(a_1,\ldots,a_n)$.

PROOF. For brevity we shall write P_i instead of $P_i(a_1,\ldots,a_i)$. Observe that $P_{n+1}P_n = P_{n+1}P_nP_nP_n = a_{n+1}P_nP_na_{n+1}P_nP_nP_n = a_{n+1}P_nP_nP_n(a_{n+1}P_n)P_n = a_{n+1}P_n(a_{n+1}P_n)P_n = \mathbf{1}P_n = P_n$ which proves the lemma in the case $m = 1$. Suppose now that the lemma holds for m. Then $P_{n+m+1}P_n = P_{n+m+1}P_nP_nP_n = a_{n+m+1}P_{n+m}P_{n+m}a_{n+m+1}P_nP_nP_n = a_{n+m+1}P_nP_n(P_{n+m}P_nP_n)(P_{n+m}P_nP_n)(a_{n+m+1}P_nP_n)P_n = a_{n+m+1}P_nP_n\mathbf{1}(a_{n+m+1}P_nP_n)P_n = a_{n+m+1}P_nP_n(a_{n+m+1}P_nP_n)P_n = \mathbf{1}P_n = P_n$ which was to be shown. Q.E.D.

For every $n = 1,2,\ldots$ let Eh_n be the class of all equivalential algebras satisfying the following postulate:

(χ_n) $\quad P_n(a_1,\ldots,a_n) = \mathbf{1}$.

By the height $h(\mathfrak{A})$ of an equivalential algebra $\mathfrak{A}$ we mean the supremum of cardinalities of linearly ordered subalgebras of $\mathfrak{A}$. The concept of height enables us to give the following characterization of the classes Eh_n :

THEOREM 7. For every $\mathfrak{A} \in EQ$ and $n = 1,2,\ldots$, $\mathfrak{A} \in Eh_n$ iff $h(\mathfrak{A}) \leqslant n$.

PROOF. If $h(\mathfrak{A}) > n$ then one can pick an increasing sequence $a_1,\ldots,a_n$ of non-unit elements of some linearly ordered subalgebra of $\mathfrak{A}$. By LEMMA 5 it follows that for every $m = 1,\ldots,n-1$, $a_{m+1}a_m = a_m$. Now using induction with respect to m we shall verify that for every $m = 1,\ldots,n$, $P_m(a_1,\ldots,a_m) = a_m$ which means that $\mathfrak{A} \notin Eh_n$. This is trivial if $m = 1$. Suppose that $P_m(a_1,\ldots,a_m) = a_m$ for some $m < n$. Then $P_{m+1}(a_1,\ldots,a_{m+1}) = a_{m+1}P_m(a_1,\ldots,a_m)P_m(a_1,\ldots,a_m)a_{m+1} = a_{m+1}a_ma_ma_{m+1} = a_ma_ma_{m+1} = a_{m+1}$ which was to be shown. Thus we have proved that (i) implies (ii). To prove the converse implication suppose that $\mathfrak{A} \notin Eh_n$ i.e. $P_n(a_1,\ldots,a_n) \neq \mathbf{1}$ for some $a_1,\ldots,a_n \in A$. Then by LEMMA 6 it follows that $\langle\{\mathbf{1}\}\cup\{P_m(a_1,\ldots,a_m) : m = 1,\ldots,n\}, \leftrightarrow\rangle$ is a linearly ordered subalgebra of $\mathfrak{A}$ having $n+1$ elements and thus $h(\mathfrak{A}) > n$. Q.E.D.

The classes Eh_n, $n = 1,2,\ldots$ and EL correspond to equivalential fragments of well-known intermediate logics, namely to the equivalential fragments of the logics LP_n of Hosoi [6] and the equivalential fragment of the logic LC of Dummett [3]. Letting $ELh_n = EL \cap Eh_n$, $n = 1,2,\ldots$, one obtains equational classes of equivalential algebras corresponding to the equivalential fragments of the logics LC_n of Thomas [9]. By THEOREM 6 and THEOREM 7 we have the following:

COROLLARY. If $\mathfrak{A} \in EQ$ is strongly compact then $\mathfrak{A} \in ELh_n$ iff $\langle A, \leqslant_{\mathfrak{A}}\rangle$ is a chain of m elements for some $m = 1,\ldots,n$.

By the theorems stated above it is easy to see that there exist infinitely many equational subclasses of EQ. It should be noted, however, that the following problem remains open:

PROBLEM. What is the number of equational classes of equivalential algebras (some related questions are discussed in [10]).

We now come to discuss a correspondence between equivalential algebras and the equivalential fragment of INT. Recall that a partially ordered set $\mathfrak{B} = \langle B, \leqslant \rangle$ is called relatively pseudocomplemented meet-semilattice (rpms) iff for every $a,b \in B$ there exists the infimum ($\inf(a,b)$) and the pseudocomplement of a relative to b ($\mathrm{rpc}(a,b)$) i.e. the greatest element in the set $\{c : c \in B, \inf(a,c) \leqslant b\}$ (see [4]). An algebraic analogue of the concept of rpms is the following: a meet-semilattice with relative pseudocomplementation (msrp) is an algebra $\mathfrak{A} = \langle B, \wedge, \rightarrow \rangle$ with two binary operations satisfying the following postulates for every $a,b,c \in B$:

(i) $a \wedge a = a$,
(ii) $a \wedge b = b \wedge a$,
(iii) $a \wedge (b \wedge c) = (a \wedge b) \wedge c$,
(iv) $a \rightarrow a = b \rightarrow b$,
(v) $a \wedge (a \rightarrow b) = a \wedge b$,
(vi) $(a \wedge b) \rightarrow c = a \rightarrow (b \rightarrow c)$.

If $\mathfrak{A}$ is msrp then the relation $\leqslant_{\mathfrak{A}}$ such that $a \leqslant_{\mathfrak{A}} b$ iff $a \wedge b = a$ is a partial ordering of B and therefore we have the partially ordered set $\mathfrak{1}(\mathfrak{A}) = \langle B, \leqslant_{\mathfrak{A}} \rangle$ corresponding to $\mathfrak{A}$. Conversely, if $\mathfrak{B}$ is rpms then we have the corresponding algebra $\mathfrak{\Gamma}(\mathfrak{B}) = \langle B, \wedge_{\mathfrak{B}}, \rightarrow_{\mathfrak{B}} \rangle$ such that $a \wedge_{\mathfrak{B}} b = \inf(a,b)$ and $a \rightarrow_{\mathfrak{B}} b = \mathrm{rpc}(a,b)$. Recall the well-known fact:

THEOREM 8. (i) If $\mathfrak{B}$ is rpms then $\mathfrak{\Gamma}(\mathfrak{B})$ is msrp and $\mathfrak{1}(\mathfrak{\Gamma}(\mathfrak{B})) = \mathfrak{B}$,
(ii) If $\mathfrak{A}$ is msrp then $\mathfrak{1}(\mathfrak{A})$ is rpms and $\mathfrak{\Gamma}(\mathfrak{1}(\mathfrak{A})) = \mathfrak{A}$.

It is known that the concept of msrp is an algebraic counterpart of the conjunctively-implicational fragment of INT. The following definitions are suggested by the standard definition of the equivalence connective in the terms of the conjunction and the implication. Let $\mathfrak{A}$ be msrp, by E-reduct of $\mathfrak{A}$ we mean the algebra $\mathbf{E}\mathfrak{A} = \langle B, \leftrightarrow \rangle$ with a binary operation $\leftrightarrow$ such that $a \leftrightarrow b = (a \rightarrow b) \wedge (b \rightarrow a)$ for every $a,b \in B$. We say that an equivalential algebra $\mathfrak{A}$ is E-embeddable into $\mathfrak{A}$ iff $\mathfrak{A}$ is embeddable into $\mathbf{E}\mathfrak{A}$. A basic result for equivalential algebras is the following:

EMBEDDING THEOREM. Every equivalential algebra is E-embeddable into a msrp.

PROOF. Given $\mathfrak{A} \in \mathrm{EQ}$, for every filter $\Psi \in F(\mathfrak{A})$ let $\overline{\Psi} = \{\Phi : \Phi \in F(\mathfrak{A}), \Psi \subseteq \Phi\}$. A subset $R \subseteq F(\mathfrak{A})$ is called hereditary iff $R \neq \emptyset$ and for every $\Psi \in F(\mathfrak{A})$, if $\Psi \in R$ then $\overline{\Psi} \subseteq R$. Let H be the family of all hereditary subsets of $F(\mathfrak{A})$. It is obvious that H is closed under arbitrary unions and intersections and therefore $\mathfrak{H} = \langle H, \subseteq \rangle$ is rpms. By THEOREM 8(i) it follows that $\mathfrak{\Gamma}(\mathfrak{H}) = \langle H, \cap, \rightarrow \rangle$ where $R \rightarrow S = \bigcup\{T : T \in H, R \cap T \subseteq S\}$ is a msrp. For every $a \in A$ we put $F(a) = \{\Phi : \Phi \in F(\mathfrak{A}), a \in \Phi\}$. We shall prove that the mapping $a \mapsto F(a)$ is E-embedding of $\mathfrak{A}$ into $\mathfrak{\Gamma}(\mathfrak{H})$. Obviously $F(a) \in H$ for every $a \in A$. By THEOREM 4 we get that if $a \in [b)$ and $b \in [a)$ then $a = b$ which yields that our mapping is one-one and thus it remains to show that for every $a,b \in A$, $F(ab) = F(a) \leftrightarrow F(b)$ where $F(a) \leftrightarrow F(b) = (F(a) \rightarrow F(b)) \cap (F(b) \rightarrow F(a))$. The fact that $F(ab) \subseteq F(a) \leftrightarrow F(b)$ is almost obvious. To prove the converse inclusion suppose that $\Psi \in F(a) \leftrightarrow F(b)$. Then $F(a) \cap \overline{\Psi} \subseteq F(b)$ and $F(b) \cap \overline{\Psi} \subseteq F(a)$ which gives that $b \in [\Psi \cup \{a\})$ and $a \in [\Psi \cup \{b\})$. Applying THEOREM 4 we get $ab \in [\Psi)$ and thus $ab \in \Psi$ since Ψ is a filter. This means that $\Psi \in F(ab)$ which was to be shown. Q.E.D.

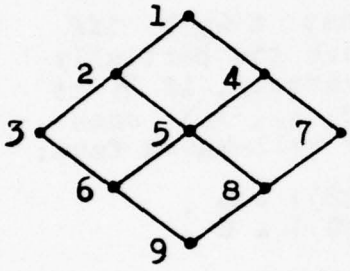
It is easy to verify that E-reduct of msrp belongs to EQ, this fact combined with EMBEDDING THEOREM yields the following:

THEOREM 9. If $\mathfrak{A}$ is an algebra with a binary operation then $\mathfrak{A} \in EQ$ iff it is isomorphic to a subalgebra of E-reduct of a msrp .

THEOREM 10. If $\mathfrak{B}$ is a free msrp free-generated by $G \subseteq B$ then the subalgebra of $E\mathfrak{B}$ generated by G is a free equivalential algebra free-generated by G.

By The well-known result of Diego [2] we know that every msrp generated by a finite set is finite. Thus THEOREM 10 yields that every equivalential algebra generated by a finite set is finite.

The free equivalential algebra free-generated by a singleton has two elements (it is isomorphic to E-reduct of the two-element Boolean algebra) and the free equivalential algebra free-generated by a dubleton has nine elements (it is isomorphic to E-reduct of the pseudo-Boolean algebra presented on the diagram below).



1 = [aa], 2 = [baab], 3 = [abb], 4 = [abba], 5 = [baab(abba)], 6 = [a], 7 = [baa], 8 = [b], 9 = [ab].

It seems to us that the number of elements of the free equivalential algebra free-generated by a three-element set can be determined by means of a computer, we have not succeed, however, in looking for a general method of computing the number of elements of the free equivalential algebra free-generated by n-element set for every natural number n .

REFERENCES

[1] G. Birkhoff, Subdirect unions in universal algebra, Bull.Amer. Math.Soc., 50(1944), pp. 764 - 768.

[2] A. Diego, Sur les algébres de Hilbert, Collection de logique mathématique, Ser. A, fasc. 21, Paris, 1966.

[3] M. Dummett, A propositional calculus with a denumerable matrix, JSL, 24(1959), pp. 97 - 106.

[4] G. Gratzer, Lattice theory, San Francisco, 1971.

[5] G. Gratzer, Universal algebra, Princeton, 1968.

[6] T. Hosoi, On intermediate logics I, J.Fac.Sci.,Univ.Tokyo Sec. 1, 14(1967), pp. 293 - 312.

[7] H. Rasiowa and R. Sikorski, The mathematics of metamathematics, Warsaw, 1963.

[8] R.E. Tax, On the intuitionistic equivalential calculus, Notre Dame Journal of Formal Logic, 14(1973), pp. 448 - 456.

[9] I. Thomas, Finite limitations of Dummett's LC, Notre Dame Journal of Formal Logic, 3(1962), pp. 170 - 174.

[10] A. Wroński, The degree of completeness of some fragments of the intuitionistic propositional logic, Reports on Mathematical Logic, 2(1974), pp. 55 - 61.

SUPERVALUATIONS IN TWO DIMENSIONS

Hans G. Herzberger
University of Toronto
Canada

1. Two Degrees of Freedom Combined

Many-valued logic has traditionally been, not only a logic of many values, but also a logic built up algebraically---with finitary operations playing a central role. This truth-functional character has often been singled out as the source of intuitive discrepancies in many-valued systems (see [1] p. 25, [8] p. 278, [10] p. 135). And these discrepancies have led in turn to the development of non-truthfunctional methods, of which supervaluations are the best known and most highly developed example (see [11], [12], [3], [4], [5]).

In pursuing the novel course of non-truthfunctional methods, supervaluational logic has at the same time reverted to an orthodox position on truth-values, limiting them to the traditional two. This limitation can be seen to be inessential, inasmuch as "many-valued supervaluations" could also be constructed, no less readily than those which are two-valued, through formation of products of families of many-valued valuations, as in [5].

Looking at the situation more generally, the two features are after all independent, and need neither be accepted together as in traditional many-valued logic, nor rejected together as in supervaluational logic. The most liberal position on the matter would result from allowing many truth-values without limitation to narrowly truth-functional interpretations of connectives.

The "two-dimensional" logic of [2] is a four-valued approach of this more flexible kind. It generalizes methods of Jaskowski [6] and Post [9] in such a way as to allow for separate representation of "classical" and "nonclassical" logical features within distinct dimensions of an overall semantic space. In such a logic, connectives can be interpreted as truthfunctional on one dimension, and as nontruthfunctional on another.

Comparison with some standard three-valued systems was sketched in [2]; in particular, it was shown how readily the ideas of Bochvar can be combined with classical logical structure within two-dimensional languages (see also [7]). Comparison with supervaluational languages was also initiated in [2] and has been further developed in [7]. It was observed in [2] that familiar supervaluational constructions can be simulated within two-dimensional languages, and it was conjectured that all supervaluational languages might turn out to be so representable. The present paper studies this representation problem, and advances a construction which is sufficiently general to map every possible supervaluational language onto some "compression-equivalent" two-dimensional language. This representation indicates the possibility of a more comprehensive theory of presuppositional languages, accommodating a richer variety of presuppositional

structures, within which the theory of [11] and [12] might be reconstructed as a special case.

2. Two-Dimensional Languages

Two-dimensional languages are a special kind of four-valued language, whose sentences are subject to two separate valuations, one on each "dimension" of a semantic space. Intuitively, the two component valuations can be thought of as registering two distinct aspects of the meaning of sentences of the language. Thus each sentence of such a language will have associated with it an ordered pair of classical two-valued propositions, somewhat in the manner of Post's construction of "higher-dimensional propositions" in [9]. This provides a way of implementing the Fregean notion that sentences can bear at least two distinct semantic relations to propositions: each sentence can be thought of as expressing one proposition, and as presupposing another. This interpretation is especially relevant for purposes of comparison with supervaluational languages.

Formally, a two-dimensional language is a pair $\langle S,W\rangle$ consisting of a set S of sentences and a collection W of ordered pairs $\langle u,v\rangle$ of two-valued valuations, each of which maps S into $\{0,1\}$. These languages are similar to product-languages constructed by the method of Jaskowski [6], with the major difference that W is allowed to be any subset of the cartesian product of its components, rather than the full cartesian product. Let the primary valuations of the language be the set:

$$W_1 = \{u: \langle u,v\rangle \in W \text{ for some } v\}$$

of first components of W, and let the secondary valuations of the language be the set:

$$W_2 = \{v: \langle u,v\rangle \in W \text{ for some } u\}$$

of second components of W. Then for two-dimensional languages in general, $W \subseteq W_1 \times W_2$, whereas a product language in the usual sense would have $W = W_1 \times W_2$. Also, W_1 and W_2 are permitted to be any collections of two-valued valuations (arbitrary subsets of 2^S), not necessarily "truth-functional" in the usual sense. Both these degrees of freedom are important for the intended interpretations of these languages.

It may be helpful to remark in a general way on connections between two-dimensional languages and the literature on four-valued logics. Two-dimensional languages as so far characterized are four-valued. This would not make them languages incorporating four-valued logics however, unless the term "four-valued logic" were extended beyond the systems in the literature to accommodate the extra degrees of freedom described above. Apart from this consideration, the principal interpretations indicate a restricted policy of designation (either one or two values will be designated). Overall then, two-dimensional languages might be described as languages incorporating a restricted kind of extended four-valued logic.

3. Languages With a Two-Fold Character

Supervaluational languages (henceforth "superlanguages") have been developed on a two-fold pattern, as languages constructed from underlying languages by oper-

ating on their admissible valuations. Taking the underlying language to be a pair $\langle S,V\rangle$ consisting of a set S of sentences and a collection V of valuations over S, a superlanguage is a triple $\langle S,V,V^*\rangle$, where each member of V^* is the supervaluation established by some nonempty collection K of underlying valuations ($K \subseteq V$), in a manner to be discussed in Section 5 below.

Every two-dimensional language can also be regarded as a two-fold construction on an underlying language, along similar lines. For comparative purposes then, the pair $\langle S,W\rangle$ can always be rewritten uniquely as the triple $\langle S,Vo,W\rangle$, with the primary valuations for W determining Vo as follows. Each $w \in W_1$ has the range $\{0,1\}$; let Rw be the isomorphic function over the range $\{F,T\}$. Then $Vo = \{Rw: w \in W_1\}$.

In their logic, two-fold languages have a correspondingly double aspect. Each has an entailment relation determined as usual by its admissible valuations, and also an underlying entailment relation determined by its underlying valuations. In the literature on superlanguages, these have been called entailment and "classical" consequence respectively. Let $L = \langle S,V,V^*\rangle$ be any superlanguage, let X be any subset of S and A be any member of S. Then X entails A in L iff $v(A) = T$ for each $v \in V^*$ that satisfies X; and A is a classical consequence of X relative to L iff $v(A) = T$ for each $v \in V$ that satisfies X.

In two-dimensional languages, the two entailment relations will be called composite and primary entailment respectively, and the defining conditions are exactly analogous once the notions of truth and satisfaction are translated into the two-dimensional framework, by identifying T with the composite value $\langle 1,1\rangle$. A two-dimensional valuation w will be said to satisfy a set X of sentences iff $w(B) = \langle 1,1\rangle$ for each $B \in X$. Now let $L = \langle S,Vo,W\rangle$ be any two-dimensional language, let X be any subset of S and A be any member of S. Then *A is a composite consequence of X in L* iff $w(A) = \langle 1,1\rangle$ for each $w \in W$ that satisfies X; and *A is a primary consequence of X in L* iff $v(A) = T$ for each $v \in Vo$ that satisfies X. These two entailment relations can also be described as the result of two alternative policies of designation: for the primary consequence relation we designate the pair of values $\{\langle 1,1\rangle, \langle 1,0\rangle\}$, and the composite consequence relation results from designating the single value $\langle 1,1\rangle$.

4. Compression

Two-dimensional languages are four-valued, and superlanguages are two-valued; to compare them we need to correlate their values. Taking account of the fact that supervaluations generally are partial functions, each supervaluation gives rise to a trichotomy of all sentences into true, false, and value undefined. Sentences of the third category are those which suffer "truth-value gaps." Two-dimensional valuations can be interpreted as determining refinements of such trichotomies, in accordance with the scheme already implicit in the discussion of Section 3:

T	F	t	f
$\langle 1,1\rangle$	$\langle 0,1\rangle$	$\langle 1,0\rangle$	$\langle 0,0\rangle$

Along with truth and falsity, denoted by capital letters, we now have two kinds of truth-value gaps, denoted by corresponding lower-case letters so as to reflect a certain analogy with classical values: in particular, so as to reflect agreement on primary coordinates.

Following this interpretation of the composite values, an operation can be defined which "compresses" any two-dimensional valuation into a corresponding partial two-valued valuation. This operation can be defined more generally over arbitrary functions; let the compression of any function w be that function w↓ determined by:

$$w\downarrow(x) = \left\{ \begin{array}{l} w(x) \text{ iff } w(x) \in \{T,F\} \\ \text{Undefined in all other cases} \end{array} \right\}$$

Provided that the range of w includes at least one value which can be identified with truth and perhaps another which can be identified with falsity, the compression of w will be its restriction to the "bivalent" part of its domain.

Let a two-dimensional language $\langle S,V,W\rangle$ be called compression-equivalent to a superlanguage $\langle S,V,V^*\rangle$ over the same sentences iff V^* exactly coincides with the collection $W\downarrow$ of all compressions of members of W.

T1. The entailment relation for any superlanguage exactly coincides with the composite entailment relation for any compression-equivalent two-dimensional language.

Proof: Let $\langle S,V,V^*\rangle$ be any superlanguage, with entailment relation E, and let $\langle S,Vo,W\rangle$ be any compression-equivalent two-dimensional language, with composite entailment relation C. By hypothesis, $V^* = \{w\downarrow : w \in W\}$, and by the definition of compressions, for any sentence $A \in S$, $w\downarrow(A) = T$ iff $w(A) = T$, and consequently for any set X of sentences from S, $w\downarrow$ satisfies X iff w satisfies X. From this it follows that $C = E$, for: [$v(A) = T$ for each $v \in V$ that satisfies X] iff [$w\downarrow(A) = T$ for each $w\downarrow \in W\downarrow$ that satisfies X] iff [$w(A) = T$ for each $w \in W$ that satisfies X].

5. Constancy Profiles

Supervaluations are based on the principle that "classical" valuations may incorporate some arbitrary value-assignments, which can be systematically discounted by registering invariants across selected classes of valuations. Whatever the principle of selection for the underlying classes, any nonempty class K of valuations can be said to confirm a sentence A iff $v(A) = T$ for each $v \in K$; can be said to falsify A iff $v(A) = F$ for each $v \in K$; and can be said to neutralize A in all other cases. Now the supervaluation established by K is a constancy profile for the class K; it is defined as the unique valuation ZK such that for any sentence A in the domain of members of K:

$$ZK(A) = \left\{ \begin{array}{l} T \text{ iff } K \text{ confirms } A \\ F \text{ iff } K \text{ falsifies } A \\ \text{Undefined iff } K \text{ neutralizes } A \end{array} \right\}$$

By this definition, the information contained in the class K of valuations is reflected in a certain partial valuation.

Two-dimensional constancy profiles can also be formed for any class K of valuations, relative to any chosen member of K which fixes a "perspective." For any $v \in K$, a two-dimensional valuation $\omega[v,K]$ will be formed so as to reflect v on its primary coordinate and K on its secondary coordinate:

$$\omega[v,K](A) = \left\{ \begin{array}{l} T \text{ iff } K \text{ confirms } A \\ F \text{ iff } K \text{ falsifies } A \\ t \text{ iff } K \text{ neutralizes } A \text{ and } v(A) = T \\ f \text{ iff } K \text{ neutralizes } A \text{ and } v(A) \neq T \end{array} \right\}$$

Such two-dimensional profiles reflect substantially the same information recorded in supervaluational profiles, with a little extra detail. More precisely:

T2. Each two-dimensional profile for any nonempty class of valuations, compresses to the supervaluational profile for that class (that is, for each $v \in K$: $\omega[v,K]\downarrow = ZK$).

This provides then a faithful and generally applicable two-dimensional representation for the supervaluation idea.

To illustrate these notions, let K be the class {v, v'}. Then K establishes one supervaluation and has two two-dimensional constancy profiles, as displayed:

	A	B	C
v	F	T	F
v'	T	T	F

Sample Class K Of Valuations

	A	B	C
ZK	-	T	F

Supervaluational Profile for K

	A	B	C
$\omega[v,K]$	f	T	F
$\omega[v',K]$	t	T	F

Two-Dimensional Profiles for K

6. The Construction

Now we are prepared to introduce a construction which maps every superlanguage L onto a unique two-dimensional language $L\uparrow$. If $L = \langle S,V,V^* \rangle$ then for each $v \in V^*$ it will be necessary to identify an appropriate establishing class K ($K \subseteq V$ and $ZK = v$), to serve as the basis for an associated two-dimensional profile. This can be done in a systematic manner by forming an the collection $Ev = \{u \in V: v \subseteq u\}$ of all initially admissible enlargements of v.

T3. Ev is always an establishing class for v.

Proof: Assume some superlanguage $\langle S,V,V^*\rangle$ and let v be any member of V^*; it is to be shown that v = ZEv. Let A be any sentence ($A \in S$). Firstly, if v(Z) = T, then u(A) = T for each u $\in V$ such that v $\subseteq$ u; so then Ev confirms A, and thereby ZEv(A) = T. Similarly, if v(A) = F then Ev falsifies A and thereby ZEv(A) = F. Suppose finally that v is undefined for A. Then v must have at least one establishing class $K \subseteq V$ such that K neutralizes A. Then u(A) $\neq T$, u'(A) $\neq F$ for some u, u' $\in K$. Since v = ZK, v $\subseteq$ u and v $\subseteq$ u', so u, u' $\in E$v. Therefore, Ev in this case neutralizes A, and ZEv is undefined for A. Since A was arbitrarily chosen, this secures that v = ZEv. Ev is not only an establishing class for v; it is also its maximal establishing class within V.

Any superlanguage $L = \langle S,V,V^*\rangle$ now has a unique two-dimensional expansion $L\uparrow = \langle S,V_o,W^*\rangle$ whose admissible valuations are determined as the collection $W^* = \{\omega[u, Ev] : v \in V^* \ \&\ u \in Ev\}$ of all two-dimensional profiles for all supervaluational establishing classes relative to the original superlanguage L.

T4. Any superlanguage is compression-equivalent to its own two-dimensional expansion.

Proof: For each v $\in V^*$, Ev is an establishing class for v, by T3. By T2, $\omega[u,Ev]$ compresses to v, for each u $\in E$v. So, for each v $\in V^*$ there is a w $\in W^*$ such that w$\downarrow$ = v. Conversely, each w $\in W^*$ has been constructed as $\omega[u,Ev]$ for some v $\in V^*$ and u $\in E$v. Therefore, $V^* = \{w\downarrow : w \in W^*\}$. Since $L\uparrow$ and I have the same set of sentences, we may conclude that they are compression-equivalent.

7. Normalization

An apparent limitation on this result is that $L\uparrow$ will not always be a two-fold language over exactly the same underlying language as the original superlanguage L. All underlying valuations for L are underlying valuations for L (that is: $V_o \subseteq V$), but the converse will not universally hold. It is of some interest then to find conditions under which the stronger condition $V_o = V$ can be guaranteed to hold.

Let a superlanguage $\langle S,V,V^*\rangle$ be called normalized iff each of its underlying valuations is "supervaluationally engaged" in the sense that it belongs to at least one establishing class for some supervaluation (that is: for each u $\in V$ there is at least one v $\in V^*$ such that v $\subseteq$ u). Any underlying valuation that is not supervaluationally engaged in this sense could simply be dropped from V without any adverse effect on the construction of supervaluations. So any superlanguage $L = \langle S,V,V^*\rangle$ can be associated with a uniquely determined normalization $NL = \langle S,V^-,V^*\rangle$, where V^- is the collection {u $\in V$: v $\subseteq$ u for at least one v $\in V^*$} of all supervaluationally engaged members of V.

The relation between any superlanguage and its normalization is one of strong equivalence in the sense of [3]. Any two languages are strongly equivalent iff they share the same sentences and admissible valuations, although they may differ in their internal construction. Applied to two-fold languages, strong equivalence permits differences in underlying valuations.

T5. The normalization of any superlanguage is always a strongly equivalent normalized superlanguage.

Proof: Let $L = \langle S, V, V^* \rangle$ be any normalized superlanguage, and let $L^\uparrow = \langle S, Vo, W^* \rangle$ be its two-dimensional expansion. Each $u \in V$ is an enlargement of at least one $v \in V^*$ (that is, $v \subseteq u$), so that $u \in Ev$. So there is some $w \in W^*$ such that $w = \omega[u, Ev]$; and thereby the set Vo of underlying valuations for $L^\uparrow$ exhausts the underlying valuations for L. So $L^\uparrow$ can be represented as the two-fold language $\langle S, V, W^* \rangle$ over the same underlying language as L.

Since every superlanguage without restriction can be normalized, and every normalized superlanguage has a perfect two-dimensional counterpart, the conjectures of [2] have now been supported in some detail. To complete the comparison between the two methods, it is readily seen that there are two-dimensional languages which are compression-equivalent to no superlanguage over the same initial language. Holding the initial language fixed then, the collection of all constructible superlanguages is strongly equivalent to a proper subset of the collection of all compressions of constructible two-dimensional languages.

REFERENCES

[1] H. G. Herzberger, "Truth and Modality in Semantically Closed Languages" in R. L. Martin, editor: The Paradox of the Liar, 1970.

[2] ----------, "Dimensions of Truth," Journal of Philosophical Logic, 1973.

[3] ----------, "Canonical Superlanguages," Journal of Philosophical Logic, in press.

[4] ----------, "Presuppositional Policies," Philosophia, in press.

[5] ----------, "The Algebra of Supervaluations," typescript, 1974.

[6] S. Jaskowski, "Recherches sur le systeme de la logique intuitioniste," 1936, translated in S. McCall, editor: Polish Logic.

[7] J. N. Martin, "Sortal Presuppositions," Ph.D. Dissertation, University of Toronto, 1973.

[8] S. McCall, "Temporal Flux," American Philosophical Quarterly, 1966.

[9] E. Post, "Introduction to a General Theory of Elementary Propositions, American Journal of Mathematics, 1921.

[10] A. Prior, Past, Present and Future, 1967.

[11] B. C. van Fraassen, "Presuppositions, Supervaluations, and Free Logic," in K. Lambert, editor: The Logical Way of Doing Things, 1969.

[12] ----------, Formal Semantics and Logic, 1971.

SIMILARITY AS A THEORY OF GRADED EQUALITY FOR A CLASS OF MANY-VALUED PREDICATE CALCULI

Charles G. Morgan
University of Alberta
Edmonton, Alberta
Canada

0. Introduction

We take as our starting point the development of the general theory of many-valued predicate calculi given by Rosser and Turquette in [2] and the development of a general theory of absolute equality for such languages given by Morgan in [1]. Although familiarity with these works will make the task of the reader easier, our treatment here is essentially self-contained.

In many-valued logics there are two distinct ways of conceptualizing the equality relation. On the one hand, we may think of equality as being in some sense absolute, analogous to the two-valued case. In this absolute sense, equality is an all-or-none relation, not capable of holding to various degrees; one item is either equal to another or it is not. A general theory of this absolute sense of equality was developed in [1]. On the other hand, we may view equality as a graded relation, analogous to the other predicates of the language, capable of holding to various degrees. In this graded sense, it is perhaps more fruitful (and accurate) to speak of a theory of similarity rather than a theory of equality, and we will henceforth adopt this terminology. It is our purpose in this paper to develop a general theory of similarity in a many-valued context.

1. Many-valued predicate calculus

In order to specify the syntax of the languages under consideration, we must first give some of the semantic presuppositions. The semantic range is taken to be some set of integers $\{1, \ldots, M\}$, for some $M > 1$. We do not assume that these values have any intrinsic meaning themselves but allow that they may perhaps be regarded as subscripts designating some other unspecified values. We assume given some "critical" integer S, $1 \leq S < M$. Statements with values i in the range $1 \leq i \leq S$ are regarded as "assertible", i.e., as in some sense true; such values are said to be "designated". Statements with values j in the range $S < j \leq M$ are not regarded as assertible but as being in some sense false; such values are said to be "undesignated". Our syntax includes the following:

1. set of n-ary relations, for n = 1, 2, ...: $R_n = \{P_1^n, P_2^n, \ldots\}$
2. set of variables $v = \{x_1, y_1, x_2, y_2, \ldots\}$
3. monadic sentence operators:
   a. negation: $\neg$

b. J operators: J_i, for integer i, $1 \leq i \leq M$

c. universal quantifiers: (x_i), for any variable in v

4. dyadic sentence operators:

a. disjunction: $\vee$

b. conjunction: &

c. conditional: $\supset$

We denote the set of all relations by $R = \bigcup_n R_n$. We assume the standard definitions of free and bound occurrences of variables, atomic formulas, and sell-formed formulas. We use the term "expression" for "well-formed formula". For convenience we assume expressions contain no vacuous quantifiers, and that all quantifiers occurring in any given expression are with respect to distinct variables. We use E, E', etc. with the above connectives as meta-expressions. Further, we use the summation symbol to denote finite disjunctions:

$$\sum_{i=m}^{n} E_i =_{df} E_m \vee \ldots \vee E_n$$

When no parentheses are included, we assume association to the left.

We further elaborate our semantics by defining the models. A model is just an ordered pair (D, V). D is any non-empty set of objects, and is called the domain of the model. V is any function which: (i) maps v into D; and (ii) maps each predicate symbol $P_i^j \in R$ to a function $\underline{P}_i^j$ from D^j into the semantic range. Further, V maps each expression into the semantic range. For atomic expressions we have:

$$V(P_i^j x_1 \ldots x_j) = \underline{P}_i^j(V(x_1), \ldots, V(x_j))$$

The mapping of the complex expressions depends on the specific semantic definition of each of the operators. We assume:

$$V(J_k(E)) = 1 \text{ , iff } V(E) = k$$

$$= M \text{ , otherwise}$$

Our only other restriction is that each of the operators listed above must satisfy so-called "standard conditions"; these conditions are the following (see [2]):

(1) $V(\neg E)$ is designated iff $V(E)$ is undesignated

(2) $V(E_1 \vee E_2)$ is designated iff either one or both of $V(E_1)$ and $V(E_2)$ are designated

(3) $V(E_1 \,\&\, E_2)$ is designated iff both $V(E_1)$ and $V(E_2)$ are designated

(4) $V(E_1 \supset E_2)$ is designated iff either $V(E_1)$ is undesignated, $V(E_2)$ is designated, or both

(5) $V((x_i)E)$ is designated iff for all functions V' differing from V for at most the argument x_i, $V'(E)$ is designated

Any other connectives desired may be employed in the language, and we assume the definition of V to be extended accordingly.

It is easy to show that each expression is assigned one and only one value in each model, and that the value assigned depends only on those free variables occurring in the expression. That is, for any expression E, if V and V' agree over the free variables in E, then $V(E) = V'(E)$. We say that an expression E is satisfied by a given model just in case $V(E)$ is designated; we say that E is falsified just in case $V(E)$ is undesignated. A set of expressions is satisfied in a given model just in case every expression in the set is satisfied in the model; the set is falsified just in case at least one member is falsified. An expression is said to be valid just in case it is satisfied by every model.

We assume to be given some axiomatic treatment of the language. We use "$\vdash E$" for "E is provable" and "$\Gamma \vdash E$" for "E is provable from the expressions in the set Γ". We use "$\Vdash E$" for "E is valid" and "$\Gamma \Vdash E$" for "E is satisfied in every model satisfying the set of expressions Γ". We assume the axiomatization is such that $\Gamma \vdash E$ iff $\Gamma \Vdash E$ (as a special case, $\vdash E$ iff $\Vdash E$). We further assume that principles corresponding to universal generalization, universal instantiation, and the deduction theorem all hold (with the standard restrictions). We note that since we have assumed standard conditions, the connectives $\neg$, &, $\vee$, and $\supset$ all behave within the system just as their two-valued analogues. In particular, if $\Gamma \vdash \neg E \supset (E' \,\&\, \neg E')$, then $\Gamma \vdash E$. A quite general scheme for producing axiomatic systems of the type required is given in [2].

2. Predicate calculus with similarity

We will use "eq" as the equality symbol and "sm" as the similarity symbol. We will use "eq" for absolute equality and "sm" for graded equality. We note that when we speak <u>intuitively</u> of a high degree of similarity between x_i and y_i, this idea is <u>syntactically</u> represented by $J_k(x_i \text{ sm } y_i)$ for low values of k (i.e., those k close to 1); and an intuitively low degree of similarity is represented by high values of k (i.e., those k close to M). Once noted, this peculiarity should not confuse the reader. We may begin our treatment of the theory of similarity by considering the normal two-valued theory of equality. The two-valued theory of equality would require the following axioms:

(a.1) $(x_1)\, x_1 \text{ eq } x_1$

(a.2) $(x_1) \ldots (x_j)(y_1) \ldots (y_j)((x_1 \text{ eq } y_1 \ \& \ \ldots \ \& \ x_j \text{ eq } y_j \ \& \ P_i^j x_1 \ldots x_j) \supset P_i^j y_1 \ldots y_j)$, for all $P_i^j \in R$

(Note that these are axioms for two-valued equality, not for absolute equality in a many-valued context.) Analogous to equality, we want to require that everything be similar to itself. In fact, we want to require that everything be similar to itself to the maximum degree possible. This consideration suggests that we begin our theory with the following axiom:

(b.1) $(x_1) \ J_1(x_1 \text{ sm } x_1)$

Let us now turn our attention to (a.2). We obviously must do more than simply replace "eq" by "sm". Since similarity can hold to varying degrees and the predicate P_i^j can hold to varying degrees, we see that our theory of similarity must be much more complex than the theory of equality. Somehow we must specify how we can obtain the degree to which the predicate holds of the y's, given the degrees of similarity between the x's and the y's and the degree to which the predicate holds of the x's. We can greatly simplify the form of the problem by noting that each instance of (a.2) can be regarded as equivalent to a set of simpler axioms of the following form:

(a.3) $(x_1) \ldots (x_n) \ldots (x_j)(y_n)((x_n \text{ eq } y_n \ \& \ P_i^j x_1 \ldots x_n \ldots x_j) \supset P_i^j x_1 \ldots y_n \ldots x_j)$, for all $P_i^j \in R$ and $n = 1, \ldots, j$

In the two-valued case, anything we can derive from (a.1) and (a.2) we could derive from (a.1) and successive uses of (a.3). We would no doubt simplify our theory of similarity if we regarded degrees of P_i^j-hood as being successively built up from degrees of similarity in an analogous fashion. We are thus led to consider the following axiom scheme:

(b.2) $(x_1) \ldots (x_n) \ldots (x_j)(y_n)((J_{k'}(x_n \text{ sm } y_n) \ \& \ J_k(P_i^j x_1 \ldots x_n \ldots x_j)) \supset J_{f(k',k)}(P_i^j x_1 \ldots y_n \ldots x_j))$, for all $k, k' \in \{1, \ldots, M\}$, for all $P_i^j \in R$, and $n = 1, \ldots, j$

Of course f(k',k) is a function specified by our semantic theory of similarity.

We must be very careful in our reading of (b.2). Suppose I know that $J_k(P_1^2 aa)$ and $J_{k'}(a \text{ sm } b)$, for some items a and b. Using (b.2) we can then infer that $J_{f(k',k)}(P_1^2 ba)$ and $J_{f(k',k)}(P_1^2 ab)$; but we can <u>not</u> infer $J_{f(k',k)}(P_1^2 bb)$. In obtaining this latter expression, two substitutions of b for a were made. We can

infer, however, $J_{f(k',f(k',k))}(P_1^2 bb)$. In this way, similarity is quite unlike equality. Generally, we can substitute as many or as few instances of equals for equals as we like, without changing truth value. But if two items are merely similar, the more substitutions we make, the greater the effect on the truth value of the resultant expression.

Applying (b.2) unrestrictedly over all predicates does lead to some counter-intuitive results. Suppose we are given J_k(a sm b). Using this expression twice in the antecedent of (b.2) we can infer $J_{f(k,k)}$(b sm b). From (b.1) we then know that $f(k,k) = 1$. Now, suppose we are given J_k(a sm b) and J_k(b sm c). Using (b.2) we could infer $J_{f(k,k)}$(a sm c); that is, we could infer J_1(a sm c). But k could have been <u>any</u> value. In particular, if a and b are as dissimilar as possible and b and c are as dissimilar as possible, then (b.2) would allow us to infer that a and c are as similar as possible! This result is extremely counter-intuitive. These considerations suggest that (b.2) should not be applied over the predicate "sm". That is, we should perhaps require for (b.2) that "P_i^j" not be "sm".

We should note that (b.2) commits us to several assumptions about similarity. First such a theory of similarity is neutral with respect to the placement of similar items in atomic expressions. For example, suppose x_1 and y_1 are similar to degree k'. Then as long as $P_i^2 x_1 x_2$ and $P_i^2 x_2 x_1$ hold to the same degree, $P_i^2 y_1 x_2$ and $P_i^2 x_2 y_1$ will hold to the same degree. This assumption seems quite plausible.

Second, such a theory of similarity is neutral with respect to the predicates involved. That is, the function f is independent of P_i^j. This assumption also seems to be quite plausible.

But third (and most important), a theory based on (b.2) assumes that knowledge of degree of similarity between x_n and y_n and knowledge of degree of P_i^j-hood with respect to x_n yields <u>precise</u> knowledge of degree of P_i^j-hood with respect to y_n. This third assumption seems to be clearly too strong. Any time we have less than complete similarity we do not infer a precise degree for the property under investigation, but only some range of degrees of the property. For example, if we are told that two ball bearings are exactly similar, then we feel justified in inferring that one is .5 mm in diameter from the fact that the other is .5 mm in diameter. But on the other hand, if we are told instead that the two are only "reasonably similar", then we feel justified only in inferring that the size of one is "reasonably close" to the size of the other. Of course the two <u>may</u> have exactly the same size and differ with respect to some other property (e.g., weight), but without further information, we have no way of knowing. These intuitions suggest that the following axiom scheme may be more appropriate:

$$(b.3)\quad (x_1)\ \ldots\ (x_n)\ \ldots\ (x_j)(y_n)((J_{k'}(x_n \text{ sm } y_n) \ \&\ J_k(P_i^j x_1 \ldots x_n \ldots x_j)) \supset \sum_{p\,=\,\text{mnf}(k',k)}^{\text{mxf}(k',k)} J_p(P_i^j x_1 \ldots y_n \ldots x_j))$$

for all k, k' $\in \{1, \ldots, M\}$, for all $P_i^j \in R$, and n = 1, ..., j

where "mxf" is the function of k and k' which gives the maximum degree of uncertainty of P_i^j-hood and "mnf" is the function of k and k' which gives the minimum degree of uncertainty of P_i^j-hood. We note that (b.3) preserves the first two assumptions discussed above for (b.2) while avoiding the objectionable third. We also note that (b.3) avoids the counter-intuitive results of (b.2) without restrictions on the predicates over which it applies.

Obviously certain restrictions must be imposed on the two functions "mnf" and "mxf". In the first place, given our semantic range, we must require that for all k and k':

$$(c.1)\quad \text{mnf}(k',k) \geq 1$$

$$(c.2)\quad \text{mxf}(k',k) \leq M$$

In addition, we always want mnf to be less than or equal to mxf; so we require that for all k and k':

$$(c.3)\quad \text{mnf}(k',k) \leq \text{mxf}(k',k)$$

Further, if we want to regard the limiting case of similarity as being like equality, we must have for all k:

$$(c.4)\quad \text{mnf}(1,k) = k$$

$$(c.5)\quad \text{mxf}(1,k) = k$$

These two conditions tell us that when two objects are as similar as they can possibly be, then they possess all the properties to exactly the same degree.

On the other hand, what can we say about the values of mnf and mxf when two objects are as dissimilar as possible? Two conflicting views seem to present themselves. The first view may be expressed by saying that similarity represents "negative correlation" as well as "positive correlation". On this view, if two objects are not very similar, we should be able to infer that if one object has a given property to a high degree, then the other has that property to a low degree; and if one object has a property to a low degree, then the other has the property to a high degree. This view of similarity deviates markedly from classical equality. If two items are not equal then classically we can infer nothing about the P-hood of the first from the fact that the second lacks P. The second view of similarity more

closely resembles classical equality. The second view may be expressed by saying that similarity represents only "positive correlation". On this view, the less similar two objects are, the less we can infer about any specific property of one object from information about the corresponding property of the other object. It is this latter view of similarity which we will treat here. Our choice is motivated not only by personal intuitions and observations of the way arguments are actually conducted, but also by the simplicity of the formal treatment which results from this view. This is not the place to debate the intuitions and observations; any justification of our theory must ultimately rest on its usefulness. This second view of similarity then suggests the following two restrictions:

(c.6) $\mathrm{mnf}(M,k) = 1$

(c.7) $\mathrm{mxf}(M,k) = M$

But in addition, this latter point of view also suggests that the less similarity between two objects, then the greater the range of uncertainty about the degree of P_i^j-hood of one that can be inferred from a given degree of P_i^j-hood of the other. In other words, it seems we should also adopt the following:

(c.8) If $k' > k''$, then $\mathrm{mnf}(k',k) \leq \mathrm{mnf}(k'',k)$ and $\mathrm{mxf}(k',k) \geq \mathrm{mxf}(k'',k)$.

In fact, we will probably want to require that "$\leq$" be "$<$" and "$\geq$" be "$>$" as long as $\mathrm{mnf}(k'',k) \neq 1$ and $\mathrm{mxf}(k'',k) \neq M$, respectively.

All of these restrictions can be met if we make the following definitions of our two functions:

(c.9) $\mathrm{mnf}(k',k) = \max\{1, k - (k' - 1)\}$

(c.10) $\mathrm{mxf}(k',k) = \min\{M, k + (k' - 1)\}$

In other words, to say that two objects x_n and y_n are similar to degree k' means that the value of any property of y_n lies within a $k' - 1$ neighborhood of the value of that property for x_n. So, for example, if $J_1(x_n \text{ sm } y_n)$, then we can infer that x_n and y_n do not differ at all in the degree to which any properties hold of them. On the other hand, if $J_M(x_n \text{ sm } y_n)$, then we can infer nothing about the degree to which any given property holds of y_n from our knowledge of the degree to which that property holds of x_n. Our proposals for mnf and mxf have much in common with a suggestion made by Scott in [3].

A strong advantage of our definitions is that they allow us to determine a unique uncertainty range for successive instantiations of similar objects, regard-

less of the order of instantiation. For example, suppose we are given the following information:

$$(d.1)\quad J_k(P_1^2 x_1 x_2)$$

$$(d.2)\quad J_{k'}(x_1 \text{ sm } y_1)$$

$$(d.3)\quad J_{k''}(x_2 \text{ sm } y_2)$$

We will assume (c.9) and (c.10) from this point on. Using (b.3) first with (d.1) and (d.2) we can obtain:

$$(d.4)\quad \sum_{p = \max\{1,\ k-(k'-1)\}}^{\min\{M,\ k+(k'-1)\}} J_p(P_1^2 y_1 x_2)$$

Then using (b.3) with (d.4) and (d.3) we obtain:

$$(d.5)\quad \sum_{p = \max\{1,\ k-(k'-1)\}}^{\min\{M,\ k+(k'-1)\}} \sum_{q = \max\{1,\ p-(k''-1)\}}^{\min\{M,\ p+(k''-1)\}} J_q(P_1^2 y_1 y_2)$$

But (d.5) is just:

$$(d.6)\quad \sum_{p = \max\{1,\ k-((k'+k'')-2)\}}^{\min\{M,\ k+((k'+k'')-2)\}} J_p(P_1^2 y_1 y_2)$$

Now, using (b.3) first with (d.1) and (d.3) we can obtain:

$$(d.7)\quad \sum_{p = \max\{1,\ k-(k''-1)\}}^{\min\{M,\ k+(k''-1)\}} J_p(P_1^2 x_1 y_2)$$

Using (b.3) with (d.7) and (d.2) we obtain:

$$(d.8)\quad \sum_{p = \max\{1,\ k-(k''-1)\}}^{\min\{M,\ k+(k''-1)\}} \sum_{q = \max\{1,\ p-(k'-1)\}}^{\min\{M,\ p+(k'-1)\}} J_q(P_1^2 y_1 y_2)$$

But (d.8) is just (d.6).

We should be careful to point out, however, that (b.3) is stated for atomic expressions only. Our definitions of "mnf" and "mxf" hold for atomic expressions only. This fact marks a deviation from classical equality. Although the corresponding axiom for classical equality, manely (a.2), is also stated for atomic expressions only, we can always prove a generalization which allows substitution of equals

for equals in any complex expression. As long as two items are similar to degree 1, we can prove a corresponding theorem for complex expressions in the many-valued case. But when the degree of similarity is not 1, then for complex expressions we must calculate the deviation in truth values, occasioned by substitution, from the deviation in truth values of the atomic constituents. This departure from classical equality is intuitively quite sound. Suppose I know that some complex expression ϕ holds of item b and that I also know that b and c are somewhat similar. The more complex ϕ is, the more likely it seems that ϕ marks a distinction between b and c, and so the less certain we are that ϕ holds of c.

To take just one revealing example, suppose that items b and c differ by 1 point in the degree to which they possess P^1_1-hood, and that both $J_2(b \text{ sm } c)$ and $J_1(J_k(P^1_1 b))$ are designated. Clearly both $J_1(J_k(P^1_1 c))$ and $J_2(J_k(P^1_1 c))$ are undesignated (assuming $M > 2$), and thus their disjunction is undesignated. So if we adopted a generalization of (b.3) to all expressions, we would be able to infer undesignated conclusions from designated premises! To put the point in a slightly different way, if similarity is to be truly a <u>graded</u> theory of equality, then the <u>only</u> definitions for "mnf" and "mxf" which generalize over all complex expressions are the constant functions 1 and M, respectively. Obviously, such a theory would give us no information at all.

Assuming that our syntax is complete with respect to the desired semantics, appropriate generalizations of (b.3) for each complex expression will be derivable. The maximum and minimum deviations in truth value for a given complex expression occasioned by similarity substitution will be dependent on the particular semantic definitions of the connectives occurring in the expression.

The proposed definitions of mnf and mxf do not however guarantee the symmetry of the sm predicate. We surely want to require the following:

(b.4) $(x_1)(y_1)(J_k(x_1 \text{ sm } y_1) \supset J_k(y_1 \text{ sm } x_1))$, for $k \in \{1, \ldots, M\}$

But even assuming (c.9) and (c.10), using (b.1) and (b.3), from $J_k(x_1 \text{ sm } y_1)$ we would only be able to deduce $J_1(y_1 \text{ sm } x_1) \vee \ldots \vee J_k(y_1 \text{ sm } x_1)$. Hence (b.4) must be adopted as an axiom.

Our axiomatic development is then composed of (b.1), all instances of (b.4), and all instances of:

(b.5) $(x_1) \ldots (x_n) \ldots (x_j)(y_n)((J_{k'}(x_n \text{ sm } y_n) \;\&\; J_k(P^j_i x_1 \ldots x_n \ldots x_j)) \supset$

$$\sum_{p = \max\{1,\, k-(k'-1)\}}^{\min\{M,\, k+(k'-1)\}} J_p(P^j_i x_1 \ldots y_n \ldots x_j))$$

for all $k, k' \in \{1, \ldots, M\}$, for all $P^j_i \in R$, and $n = 1, \ldots, j$

A language $\underline{L}$ will be said to be "with similarity" if R_2 is not empty and there is a specific element sm of R_2 called the similarity symbol. By SML (similarity axioms for $\underline{L}$) we mean the set of expressions consisting of (b.1), all instances of (b.4), and all instances of (b.5).

3. Completeness

We must now describe the set of "intended" models and prove that our axioms are complete with respect to those models. The intended models will be called sm-normal (similarity-normal) models. In any model, the similarity symbol sm must be interpreted as a function $\underline{sm}$ from D^2 into the semantic range $\{1, \ldots, M\}$. Our definition of sm-normal model is given in terms of restrictions on the function $\underline{sm}$.

Let a model (D,V) be given. For any ordered j-tuple $(d_1, \ldots, d_j)$ of objects, not necessarily distinct, we mean by the permutation set $\langle d_1, \ldots, d_j \rangle$ the set of all possible ordered j-tuples obtained from $(d_1, \ldots, d_j)$ by permuting the objects. Obviously if one j-tuple $(d_1, \ldots, d_j)$ is just a permutation of another j-tuple $(d_1', \ldots, d_j')$, then the same permutation set is obtained from each. We denote arbitrary j-tuples by (j) and (j)'. Given two arbitrary objects d and d' and two j-tuples (j) and (j)', we say that (j) and (j)' are correlated with respect to d and d' just in case the only difference between (j) and (j)' is that (j) has an occurrence of d in exactly one position where (j)' has an occurrence of d', providing $d \neq d'$; if $d = d'$, then (j) and (j)' are correlated just in case they are identical. We write cor(d,d',(j),(j)') as an abbreviation. We define the maximum difference function "mxdf" from $D^2 \times R$ into the semantic range as follows:

$$\text{mxdf}(d,d',P_i^j) = \max\{|\underline{P}_i^j(j) - \underline{P}_i^j(j)'| : (j) \in \langle d, d_1, \ldots, d_{j-1} \rangle, \text{cor}(d,d',(j),(j)'), \text{ and } d_1, \ldots, d_{j-1} \in D\}$$

We say that the model (D,V) is sm-normal just in case for all $d, d' \in D$:

(e.1) $\underline{sm}(d,d) = 1$

(e.2) $\underline{sm}(d,d') = \underline{sm}(d',d)$

(e.3) $\underline{sm}(d,d') \geq 1 + \max\{\text{mxdf}(d,d',P_i^j) : P_i^j \in R\}$

Intuitively, we want sm to be interpreted in such a way that the maximum absolute difference in the P_i^j values of any two objects is always in the uncertainty range

specified by (c.9) and (c.10), so (e.3) is required. Conditions (e.1) and (e.2) simply reflect in the semantics the intuitions which produced (b.1) and (b.4).

If an expression E is satisfied in every sm-normal model, we will write $\Vdash_{sm} E$. If an expressions E is satisfied in every sm-normal model which satisfies all expressions in the set Γ, then we write $\Gamma \Vdash_{sm} E$. The aim of our completeness proof is to show that if $\Gamma \Vdash_{sm} E$ then SML $\cup\, \Gamma \vdash E$. We first prove several useful preliminary results.

Theorem 1: The set of expressions SML is satisfied by every sm-normal model.

Proof: Let (D,V) be an arbitrary sm-normal model. Clearly (e.1) and (e.2) guarantee the satisfaction of (b.1) and (b.4), respectively. Consider an arbitrary instantiation of some instance of (b.5), and suppose:

$$V(x_n \text{ sm } y_n) = k'$$

$$V(P_i^j x_1 \ldots x_n \ldots x_j) = k$$

$$V(P_i^j x_1 \ldots y_n \ldots x_j) = k''$$

By definition, the two j-tuples $(V(x_1), \ldots, V(x_n), \ldots, V(x_j))$ and $(V(x_1), \ldots, V(y_n), \ldots, V(x_j))$ are correlated with respect to $V(x_n)$ and $V(y_n)$. Hence:

$$|k - k''| \leq mxdf(V(x_n), V(y_n), P_i^j) \leq \underline{sm}(V(x_n), V(y_n)) - 1 \leq k' - 1$$

Thus (b.5) must be satisfied. Q.E.D.

Theorem 2: Any model (D,V) which satisfies all expressions in SML must be sm-normal.

Proof: Suppose (D,V) satisfies all expressions in SML. Trivially the satisfaction of (b.1) guarantees the truth of (e.1). And likewise the satisfaction of all instances of (b.4) guarantees the truth of (e.2). Thus we need only show that (e.3) must hold. Our proof is by contradiction. Suppose that for some d, d' $\in$ D the following is true:

$$\underline{sm}(d,d') < 1 + \max\{mxdf(d,d',P_i^j): \; P_i^j \in R\}$$

Let us designate by P_r^s a member of R such that for all $P_i^j \in R$:

$$mxdf(d,d',P_r^s) \geq mxdf(d,d',P_i^j)$$

Thus for some s-tuples (s) and (s)' correlated with respect to d and d', we have:

$$|\underline{P}_r^s(s) - \underline{P}_r^s(s)'| > \underline{sm}(d,d') - 1$$

Let V' be any valuation just like V except perhaps:

$$V'(x_1) = d_1, \ldots, V'(x_n) = d, \ldots, V'(x_s) = d_s, V'(y_n) = d'$$

where $(s) = (d_1, \ldots, d, \ldots, d_s)$. Let k be $V'(P^s_r x_1 \ldots x_n \ldots x_s)$, let k' be $\underline{sm}(d,d')$, and let k" be $V'(P^s_r x_1 \ldots y_n \ldots x_s)$. Then by assumption, $|k - k''| > k' - 1$. But this means that one of the following must hold:

$$k'' < k - (k' - 1), \text{ or } k'' > k + (k' - 1)$$

So, given that "&", "$\supset$", and the universal quantifiers all satisfy standard conditions, V falsifies an instance of (b.5) which uses the predicate P^s_r. But this contradicts our assumption that all expressions in SML are satisfied in (D,V). Q.E.D.

A set of expressions Γ is said to be consistent iff there is no expression E such that $\Gamma \vdash E \,\&\, \neg E$. Rosser and Turquette prove both parts of the following theorem in [2] in the course of their completeness proof:

Theorem 3: A set of expressions Γ is consistent iff Γ is satisfiable.

We can use Theorem 3 with the results established above to prove both the consistency and completeness of our axioms.

Theorem 4: The set of expressions SML is consistent.

Proof: Consider any model having only one object d in the domain, with $\underline{sm}(d,d) = 1$. It is easy to verify that such a model is sm-normal. Hence by Theorem 1, SML is satisfied in the model. But then by Theorem 3, SML is consistent. Q.E.D.

Theorem 5: (Completeness) For any set of expressions Γ, if $\Gamma \mid\!\vdash_{sm} E$, then $SML \cup \Gamma \vdash E$.

Proof: Assume $\Gamma \mid\!\vdash_{sm} E$; we must show that $SML \cup \Gamma \vdash E$. We first show by contradiction that $SML \cup \Gamma \cup \{\neg E\}$ is not satisfiable. Suppose some model (D,V) satisfies $SML \cup \Gamma \cup \{\neg E\}$. Since (D,V) satisfies SML, (D,V) must be sm-normal by Theorem 2. And since (D,V) satisfies $\neg E$, it must falsify E. But this contradicts our assumption that $\Gamma \mid\!\vdash_{sm} E$, and hence $SML \cup \Gamma \cup \{\neg E\}$ is not satisfiable. Then by Theorem 3 and the definition of consistency, there must be some expression E' such that $SML \cup \Gamma \cup \{\neg E\} \vdash E' \,\&\, \neg E'$. Since the connectives satisfy standard conditions, there is no loss of generality in assuming E' to be closed. So by the deduction theorem, $SML \cup \Gamma \vdash \neg E \supset (E' \,\&\, \neg E')$; and hence $SML \cup \Gamma \vdash E$. Q.E.D.

4. Similarity and absolute equality

We have developed our general theory of similarity in part by attempting to generalize the two-valued theory of equality. The intuitively simplest generaliza-

tion of two-valued equality is the theory of absolute equality developed in [1]. A comparison of the absolute theory of equality with the theory of similarity seems in order. The absolute theory of equality for languages of the sort discussed here is composed of the following:

(f.1) $(x_1)\ x_1 \text{ eq } x_1$

(f.2) $(x_1) \ldots (x_j)(y_1) \ldots (y_j)((x_1 \text{ eq } y_1 \ \& \ \ldots \ \& \ x_j \text{ eq } y_j \ \& \ J_k(P_i^j x_1 \ldots x_j)) \supset J_k(P_i^j y_1 \ldots y_j))$, for all $P_i^j \in R$ and all $k \in \{1, \ldots, M\}$

(f.3) $(x_1)(x_2)(J_1(x_1 \text{ eq } x_2) \vee J_M(x_1 \text{ eq } x_2))$

It is not difficult to see that the equality relation so defined is a similarity relation of the sort discussed above. By (b.1)', (b.4)', and (b.5)', we mean the axioms obtained from (b.1), (b.4), and (b.5), respectively by replacing "sm" by "eq". From (f.3) and (f.1) we can derive (b.1)'. From (f.1) and (f.2) we can derive:

(f.4) $(x_1)(x_2)(x_1 \text{ eq } x_2 \supset x_2 \text{ eq } x_1)$

From (f.4) and (f.3) we can derive all instances of (b.4)'. From (f.2) and (f.1) we can derive:

(f.5) $(x_1) \ldots (x_n) \ldots (x_j)(y_n)((x_n \text{ eq } y_n \ \& \ J_k(P_i^j x_1 \ldots x_n \ldots x_j)) \supset J_k(P_i^j x_1 \ldots y_n \ldots x_j))$

And from (f.5) and (f.3) we can derive all instances of (b.5)'. Hence "eq" is a similarity relation, though of course a very special one.

Perhaps more important than the fact that absolute equality may be regarded as a similarity relation is the fact that equality may be <u>defined</u> in any language with similarity. We can define the equality symbol as follows:

(g.1) $x_i \text{ eq } x_j =_{df} J_1(x_i \text{ sm } x_j)$

Using this definition for absolute equality, it is possible to derive (f.1) - (f.3) from the axioms (b.1), (b.4), and (b.5) for similarity. Axiom (f.1) is just (b.1) by definition. Axiom (f.3) follows from the definition of the J operators and completeness. Axiom (f.2) follows by successive applications of (b.5). In this sense then equality in the absolute sense is just the simplest possible case of similarity, and the theory of similarity as we have developed it is a genuine generalization of the theory of absolute equality. Thus similarity may properly be regarded as graded equality.

5. Extensions

Our treatment of similarity has been with respect to all the predicates of the language. We have not here discussed similarity over a limited range of predi-

cates. For example, we may say that two individuals are similar in their physical characteristics, though quite unlike each other in their behavior. One way to handle such discourse would be to consider only sublanguages--for example, the sublanguage of physical characteristics. Within a given isolated sublanguage, our treatment of similarity would be quite adequate. A slightly more general approach would be to include many similarity predicates in the language, each predicate covering some subset of the predicates. We would then include a separate set of axioms for each similarity predicate. In each case, axiom (b.5) would be modified to cover only the set of predicates appropriate for each similarity predicate. If some predicates in the language did not occur in a subset for which there was a similarity predicate, then absolute equality could not be defined. But as long as there was some finite set of similarity predicates such that each predicate in the language was in the subset covered by one of the similarity predicates, absolute equality could be defined by a conjunction of expressions $J_1(x_i\ sm_k\ x_j)$ for each similarity predicate sm_k.

We have also not discussed the inclusion of function symbols. In the two-valued case, we generally include an axiom like the following:

$$(h.1)\quad (x_1) \ldots (x_n)(y_1) \ldots (y_n)((x_1 \text{ eq } y_1 \,\&\, \ldots \,\&\, x_n \text{ eq } y_n) \supset f_i^n(x_1, \ldots, x_n) \text{ eq } f_i^n(y_1, \ldots, y_n))$$

A many-valued version of (h.1) could be introduced similar to (b.5). However, we personally doubt the usefulness of doing so. Unless two items are identical, it is always possible to define a function mapping the two items into objects as dissimilar as desired. Thus it would be virtually impossible in any practical case to define non-trivial maximum and minimum deviation limits for functional substitution.

REFERENCES

[1] Morgan, C. G. "A Theory of Equality for a Class of Many-valued Predicate Calculi," Zeitschrift für mathematische Logik und Grundlagen der Mathematik, forthcoming.

[2] Rosser, J. B., and Turquette, A. R. Many-valued Logics. North-Holland Publishing Company, Amsterdam (1952).

[3] Scott, D. S. "Background to Formalization," Truth, Syntax and Modality, ed. by Hugues Leblanc. North-Holland Publishing Company, Amsterdam (1973).

ON THE NAVYA-NYĀYA LOGIC OF PROPERTY & LOCATION

Bimal Krishna Matilal
Toronto
Canada

A Synopsis of the paper:

A proposition in Navya-nyāya is analyzed in terms of property and location. Negation is always construed as term-negation. Sentential negation is usually avoided. Negation of a property generates another (negative) property. A negative statement is analyzed as predication of a negative property.

The universe U is peopled with loci or locations where properties are locatable. The presence-range of a property is the set of loci where it is locatable. The absence-range is the set of loci where it is not locatable.

A property with an empty presence-range is unlocatable. It is ruled as fictitious (e.g., golden mountain). Properties with empty absence-ranges are admitted as real (non-fictitious), e.g. knowability. They are called ever-present. Both the fictitious (unlocatables) and the ever-present are ruled as unnegatable, for negation of them does not generate real (locatable) properties.

Most properties are wholly locatable such that they are not co-locatable with their absences in the same set of loci. But some properties are partially locatable, such as chair-contact. Such a property is apparently co-locatable with its absence in the same locus. This offends the law of negation. Thus, a device is used to reparse the partially locatable properties as wholly locatable so that the standard notion of negation may not be 'mutilated' in this system.

Non-deviation and pervasion are two important logical relations in the system which help inference. The Navya-nyāya formulation of these relations are given. Navya-nyāya insistence on the non-emptiness of the presence-range of properties serves the purpose of making the existential import of general statements explicit. In this respect, non-deviation can be contrasted with the A-relation of Aristotle á la Strawson.

To explain the notion of the unnegatable as well as the negation of the partially locatable, some concepts of a multiple-valued system may be used with an entirely different interpretation of the values. The negation matrix has been given.

Despite the peculiarities mentioned above, Navya-nyāya tries to work with the standard notion of negation in a two-valued logic.

ON THE NAVYA-NYĀYA LOGIC OF PROPERTY AND LOCATION

Bimal Krishna Matilal
Toronto
Canada

The quest for good reasons which generate dependable and acceptable conclusion is almost universal. Indian logic, by which I mean a combined tradition of the Buddhist, Nyāya and the Jaina, is only another instantiation of this universal quest in the intellectual history of mankind. It represents an independent tradition of studying inference and its validity. Just because of its difference as well as independence from the Western tradition, the inference theory developed here should prove extremely interesting for both logicians and philosophers. The Indian theory of logic shows also a continuous development from the pre-Christian era upto the seventeenth century A.D. It lacks, it is true, some of the familiar logical (and mathematical machinary which logicians of today have come to expect. But then it offers a contrast in these areas with Western logical theories that developed primarily during the last two centuries. It is also instructive. For it shows, at least, what other ways are left to us for solving some logical problems in case certain familiar devices were not available.

With this as a prelude let me describe some features of what has been called "Navya-nyāya logic" or just "Navya-nyāya" - a system that developed within the Nyāya tradition beginning from 1100 A.D. It absorbed the Buddhist criticism of the earlier Nyāya school and reformulated its older theory of inference. This is an exploratory paper. In what follows, I shall first outline the Navya-nyāya concept of property and location and logical relations formulated in terms of property and location. I shall then make some observations to show the relevance of some Navya-nyāya theories to our modern concerns in the philosophy of logic.

Cognitive states:

Navya-nyāya analyses propositions in terms of property and location or locus. More correctly, Navya-nyāya analyses what I have called elsewhere judgemental or qualificative cognitive states in terms of qualifiers and qualificands (Matilal, 1968,p.12). Such a cognitive state is usually represented by a sentence. Because of the use of the term "cognitive" or "cognition" here, a logician trained in the tradition of Frege and Carnap may immediately bring the charge of "psychologism" against Navya-nyāya. But I have argued elsewhere that this is not so (Matilal, 1968, p.10). Navya-nyāya is concerned with the 'objective' content of a cognitive state and analyses the sentence that is supposed to represent the structure of such a content. It is not concerned with the psychological act of cognition as such. Thus, in Navya-nyāya logic when one cognitive state is said to be contradictory to another, it is not just their psychological impossibility that is appealed to. Rather it is asserted that to suppose that somebody should believe (or, cognise) both p and not-p at the same time is self-contradictory. In other words, what is appealed to here is the impossibility that is completely determined by the logical relation between p and not-p.

Diṅnāga (the Buddhist logician of c. 5th century A.D.) suggested a dharma-dharmin ("property and locus") analysis of a qualificative (judgemental) cognitive state. In Diṅnāga's terminology, however, such a cognitive state is called 'constructive'; for Diṅnāga, like B. Russell and British empiricists, emphasized a distinction between the data (the immediately 'given' in consciousness) and the constructs based on the data. Existence or reality is ascribed only to the data (svalakṣaṇa "unique particular"), and the constructs are products of imagination (kalpanā). Navya-nyāya rejected this ontology of data of the Buddhists, but accepted the dharma-dharmin analysis of a cognitive state or proposition.

Properties

A cognitive state is usually said to locate a property in a locus: the form is "x has p' or 'p (is) in x'. Simple predicate formulations such as 'x is F' are noted, but only to be rephrased as 'x has F-ness' (where "F-ness" stands for the property derived from "F"). Thus, we have here two sorts of individuals - properties and locations or loci. Correspondingly, we can talk about two sorts of individual constants: property-terms (r, s, t, u, w, h ....) and location-terms (l, m, n, o, p). The best example of a property-term is "blue-colour" which is locatable in a cup that is blue, the property expressed by "cowness" that is locatable in a cow (in any cow). Such physical materials as a cup, fire, smoke, water, and a pot are also treated in Navya-nyāya as properties inasmuch as they are locatable in such loci as a table, a mountain, ground, the kitchen and the plate. Hence terms expressing such physical materials are treated as property-terms. The apparent oddity of treating such things as properties can be resolved if we conceive anything to be a property that purports to have a location.

It may further be noted that even a relation may sometimes be treated as a property in Navya-nyāya. If a relation is tied in one end to the relatum, then the whole complex can be treated as a property of the other relatum. Thus, the cup-contact in the case of a cup being placed on a table can be treated as a property of that table.

Negation:

Navya-nyāya recognizes basically two types of negation: absence and difference. Most peculiar features of Navya-nyāya emerge in connection with its interpretation of negation of properties. Sentential negation is usually avoided. A negation is construed as a term-negation in either of the following ways. We get an absence when it is a negation of location, a difference when it is a negation of identity. When a negation or a negative statement negates location of a property in a locus, it is construed as ascribing the absence of that property to that locus. Thus, absence of a property is treated as another property. "The pot is not blue" is first rephrased here as "the pot does not have blue colour" which is further rephrased as "the pot has the absence of blue colour". Using the complement sign '~' for term-negation, we can represent the above statement (m = the pot, s = blue colour):

"m has ~s".

When a statement negates an identity between, say, a table and a cup, it is construed as "a table is different from a cup" ("s ≠ t"). But Navya-nyāya argues that to say that a table is different from a cup is equivalent to saying "a table lacks the essence of a cup, or simply, lacks cupness." In other words, 'difference from a cup' is said to be extensionally equivalent to 'the absence of cupness' (which means that both these properties are locatable in the same set of loci).

World of loci: Presence-range and Absence-range

Let us conceive of a universe U, which is peopled with loci or locations. Locations are called locations because they accomodate properties. And similarly, properties are properties as long as they are locatable in some locus or other.

Given a particular property t, we can find a set of locations or loci where t is locatable or present, and another set of loci where t is not locatable. Let us call the former set the presence-range of t, and the latter the absence-range of t. Let us use the notation 't+" for the presence-range of t, and 't-' for the absence-range of t. Thus, ordinarily, the two sets, t+ and t-, are supposed to exhaust the universe of loci U.

The unlocatables:

Navya-nyāya demands that the presence-range of a non-fictitious (real) preperty should be non-empty. Navya-nyāya argues that if the presence-range be empty then the property in question would be unlocatable! An unlocatable property is suspect in Navya-nyāya. It is regarded as a fictitious property which cannot be lacated in our universe of loci. It is called an a-prasiddha property (Ingalls: "unexampled" property, i.e., 'unestablished' , imaginary property). Navya-nyāya hesitates to perform logical operation on such a property. For example, one cannot negate such a property and there-by obtain or derive another (negative) property! Thus, we have the following restriction on negation: If s is a property with a non-empty presence-range, then by negating it we get another property, a negative property, s̄; but if s is unlocatable, it cannot even be successfully negated.

Properties in Navya-nyāya are either atomic (or 'simple') or composite. A composite property is formed out of atomic ones, and hence such a property is analysable into atomic components or 'simple' properties. A 'simple' property is regarded as fundamental. It is not analysable into components. (For more on the notion of 'simple' property, see Matilal, 1971, p.83-91).An example of simple property is: cowness. The absence of cowness is a composite property. All fictitious properties like the property of being a flying horse, that of being a unicorn, a golden mountain and the son of a barren woman, are composite properties, being analysable into a number of 'simple' properties. And, it is argued, such 'simple' components are always real properties in the sense that they are locatable in some locus or other.

The unnegatables:

If the presence-range of a property is empty, it is unlocatable. Nyāya calls such a property fictitious. What about properties whose absence-range is empty? Nyāya admits such properties as real, i.e., non-fictitious. They are called ever-present properties (cf. kevalānvayin). They are said to be locatable in all loci of U. Examples of such properties are: knowability, expressibility and provability (Matilal, 1968a).

An ever-present property is non-fictitious in Navya-nyāya, for its presence-range is non-empty (infact, the presence-range is the whole universe U). But since its absence-range is empty, Navya-nyāya regards such a property as unnegatable! In other words, just as an unlocatable property is said to be not negatable in Navya-nyāya, an ever-present property is also regarded as not negatable. For we cannot derive a real, non-fictitious (negative) property by negating an ever-present property. Thus, we have another restriction on the operation of negation: If s is an ever-present property, it is locatable (i.e., real), but it is nevertheless unnegatable.

It is obvious that the introduction of ever-present properties into the system involves many logical difficulties. Thus, some pre-Gaṅgeśa Nyāya logicians were definitely not in favour of its use. They argued that a true property should have a non-empty presence-range as well as a non-empty absence-range. If we rule the unlocatables as fictitious, we might as well rule the ever-present properties as fictitious, for both, as we have seen, cannot be successfully negated. But Gaṅgeśa (13th - 14th century A.D.) scorned this suggestion and argued that even if we do not accept such properties like knowability as non-fictitious, we cannot escape from admitting other kinds of ever-present properties. For, if we believe that each locus in the universe of loci is distinct from another, then this property, distinctness, can be construed as an ever-present property (for more on this argument, see Matilal, 1968a,p.533).

Sondaḍa (an opponent of Gaṅgeśa) disputed the position that the unlocatables be unnegatable. If we admit an ever-present as real (non-fictitious), i.e., accept a property to be real which is locatable in all loci, then, one might argue, by negating a so-called unlocatable property we obtain only a negative property which should be locatable in all loci. In other words, such a negative property has to be admitted as real because its presence-range is non-empty (it is only an ever-present property). Thus, if the property of being a golden mountain is unlocatable then the absence of such a property is to be located everywhere! For, it makes perfect sense to say that there is no golden mountain, or that all loci lack the property of being a golden mountain.

But Gaṅgeśa refuted Sondaḍa's contention. An unlocatable property, according to Gaṅgeśa, resists the operation of negation. Negation is restricted to the locatables and only such locatables whose absence-ranges are non-empty. To say, "there is no golden moun-

tain" means, for Gaṅgeśa, that no mountain is golden, i.e., made of gold. But 'golden mountain' as a composite property is unnegatable.

Partial location:

We face a further oddity about negation when Navya-nyāya introduces the notion of partial location (cf. avyāpya-vṛtti, Ingalls: "incomplete occurrence") of properties. Most properties are wholly or pervasively occurrent or locatable in their loci, but some properties are said to be only partially or non-pervasively occurrent or locatable in their loci.

To explain this notion we have to develop some further logical vocabularies. Let us use a descriptive predicate (a relational term), 'L' for "locus of"; we can then define some other (logical) predicates interm of this 'L'. First let us define the relation of co-location, 'C'. We can say that s is co-located with t provided there is a locus where both s and t are locatable. Thus, co-location is a symmetrical relation. In other words, one property is co-locatable with another just in case their presence-ranges intersect or overlap. Using the convention of Boole, we can say that s is co-locatable with t provided the logical product of s+ and t+ is non-empty.

In the above we have noted that if s is a locatable property then s+ and s- exhaust the universe of loci U. But we have not required that the presence-range and the absence-range of s be disjoint. In other words, we have left open the possibility of one intersecting the other. And according to Navya-nyāya conception of negation, this is not impossible! In other words, in the same locus a property and its absence may both be locatable. Navya-nyāya calls such properties partially or non-pervasively locatable.

A property is pervasively (wholly) locatable provided it is not colocatable with its absence. But when a property is co-locatable with its absence, it is called a partially locatable property. To put it in another way, if the absence-range of a property overlaps or intersects its presence-range, it is only partially locatable.

Physical contact is the best example of a partially locatable property. When I am sitting on a chair, there are places in the chair where my contact is absent. Thus, the same chair is said to be the locus of my contact (as a property) and also the absence of my contact. Obviously it impinges upon our general notion of contradiction to say that the same locus is characterized by a property and its absence at the same time. (Remember that absence of a property means only the negation of that property. How can we affirm and negate the same property of the same locus?) Thus, this doctrine of partial location requires some reformulation of the formal law of non-contradiction or of excluded middle.

An example may illustrate some further problems involved in the notion of partial location. Suppose, w is a partially locatable property. Now the absence-range of w will include not only those loci where w is absent (wholly) but also those loci where w is present. In other words, the presence-range of w̄ includes the

the presence-range of $\underline{w}$. Thus, the presence-range of $\underline{\tilde{w}}$ is the whole universe of loci, U. This means that if $\underline{w}$ is a partially locatable property then $\underline{\tilde{w}}$ is an ever-present property, for the formal character of an ever-present property will undoubtedly apply to it. Now, if we negate further $\underline{\tilde{w}}$, we are supposed to derive an unlocatable property. (Remember negation of the ever-present generate the unlocatables.) But Navya-nyāya accepts the law of double negation. Udayana (11th-12th century A.D.) formulated the law as follows: the negation of the negation of a property is identical with the property itself (Nyāya-kusumāñjali, Ch. 3). Thus, we must have: the absence of $\tilde{w}$ = $\underline{w}$. We face here an apparently paradoxical situation: if $\underline{w}$ is a partially locatable property, then $\underline{w}$ can be shown to unlocatable!

Gaṅgeśa avoids this apparent problem by pointing out that there are two kinds of ever-present property, one of which is to be treated as unnegatable but the other is negatable. It is all right to say that when $\underline{w}$ is partially locatable $\underline{\tilde{w}}$ becomes an ever-present property in the above manner. But $\underline{\tilde{w}}$ is also partially locatable with regard to some of its loci. In other words, the presence-range of $\underline{\tilde{w}}$ is actually a combination of two: its pure presence-range (where $\underline{w}$ is not present) and a mixed range where $\underline{\tilde{w}}$ is co-locatable with $\underline{w}$. Thus, $\underline{\tilde{w}}$ is a partially locatable ever-present property, and as such it is negatable. The absence-range of $\underline{\tilde{w}}$ is non-empty; it coincides with the presence-range (which is a mixed range) of $\underline{w}$. Thus,we have a formal restriction on the former restriction of negation: not all ever-present properties are unnegatable.

Gaṅgeśa saved the law of double negation by resolving the oddity in the above manner. Some Navya-nyāya writers differed from him in this regard. Raghunātha (16th century A.D.), for example, suggested that the law of double negation be given up, for it is based upon only extensional identity (their presence-ranges and absence-ranges being equal). Intensionally, $\underline{w}$ and the absence of $\underline{\tilde{w}}$ are distinguishable.

Mathurānātha (16th century A.D.) suggested a different method of resolving the above oddity. According to him, instead of treating $\underline{\tilde{w}}$ as ever-present, we should treat the expression "$\underline{\tilde{w}}$" as ambiguously referring to two distinct (negative) properties: one that is partially locatable in its loci, the other wholly locatable in its loci. The presence-range of the first is disjoint from that of the second. The first is actually co-locatable with $\underline{w}$, but the second is locatable where and only where $\underline{w}$ is not locatable. Thus, the problem of negating an ever-present property will not arise in this case.

Deviation and Pervasion:

In the above we have defined co-location. Let us define some more logical predicates such as deviation ($\underline{D}$), non-deviation ($\underline{\tilde{D}}$) and pervasion ($\underline{V}$). We can say that $\underline{h}$ deviates from $\underline{s}$ just in case the absence-range of the latter overlaps (intersects) the presence-range of the former(cf. sādhyābhāvavad-vṛttitvaṃ vyabhicāraḥ). Using Boolean convention, we can write:

$$\underline{hDs} \quad \text{iff.} \quad \underline{h}+.\underline{s}- \neq 0.$$

Similarly, h non-deviates from s if and only if s- does not overlap h+ (sādhyābhāvavad-avṛttitvam avyabhicāraḥ):

$$\underline{hNs} \quad \text{iff.} \quad \underline{h}+.\underline{s}- \not\neq 0.$$

The relation of pervasion (vyāpti) is an important relation in Navya-nyāya since it allows valid inference of one property from another. Thus, if h is pervaded by s then from the presence of h in a particular locus, we can validly infer presence of s in it. The rule is: (hLm . sVh) ⊃ sLm.

The relation 'pervaded by' is identifiable with non-deviation (defined above) as long as we talk of such properties whose absence-ranges are non-empty. (For we have used the absence-range of s in the above definition of non-deviation.) But if s is unnegatable, the above definition, according to Navya-nyāya, becomes inapplicable. Thus, Gaṅgeśa re-formulates the definition of pervasion which will be inclusive of pervasion between ever-present (unnegatable) properties.(cf. hetuman-niṣṭhābhāvāpratiyogi-sādhya-sāmānādhikaraṇyam vyāptiḥ): Thus, we may say: s pervades h if and only if (1) s is co-located with h AND (2) if the absence-range of any property t intersect the presence-range of h then t is non-identical with s.

$$\underline{sVh} \quad \text{iff.} \quad \underline{s}+.\underline{h}+ \neq 0 \quad \text{AND if } (\underline{t}-.\underline{h}+ \neq 0) \text{ then } \underline{t} \neq \underline{s}.$$

A further problem arises when s becomes a partially locatable property. For we have seen that by definition the presence-range and the absence-range of such a partially locatable property intersect. Thus, when s is partially locatable, its absence-range includes its presence-range, and thereby its absence-range intersects with the presence-range of h. Thus, the second component of the above definition may not be satisfied by such an s. Gaṅgeśa avoids this quandary by suggesting further qualification of the above definition:

$$\underline{sVh} \text{ iff. } \underline{s}+.\underline{h}+ \not\neq 0 \text{ AND if } (\underline{t}+.\underline{t}- \not\neq 0 \;\&\; \underline{h}+.\underline{t}- \neq 0) \text{ then } t \neq s.$$

Some observations:

I shall conclude with some general observations in which I shall try to connect the problems discussed above with the explicit concern of modern logicians. Let us note, first, that non-deviation and pervasion relation may be compared with the A-relation of Aristotle, for all three share the common logical feature, i.e., transitivity. For contrast, we may say that the Navya-nyāya formulation of non-deviation (or pervasion relation), while it is narrower in its scope, does not suffer from the same ambiguity which the A-relation of Aristotle seems to have suffered from (Strawson,p.163f).

It is often pointed out, for example, that the existential

import of the A-proposition should be assumed in order that all the laws of the traditional (Aristotelian) system might be satisfied. Strawson has discussed three possible interpretations of the four propositions of Aristotle, and shown that all the traditional laws can be satisfied under the third. In the context of Indian logic, we are primarily concerned with a general (affirmative) proposition which is used as the major premise. Richard S.Y.Chi has rightly pointed out (against the common misinterpretation of many modern writers on Indian logic) that the 'exemplified major' in the Indian variety of syllogism is actually to be interpreted as 'an existential major premise' (p.xxx-xxxi). By 'an existential major premise' Chi has obviously meant a general affirmative proposition where the non-emptiness of the class denoted by the subject term is presupposed.

The contrast between non-deviation (or pervasion) on the one hand and the A-relation of Aristotle on the other can be brought about in the following way. Navya-nyāya says that non-deviation of $\underline{h}$ from $\underline{s}$ holds when the following conditions are satisfied:

i) $\underline{h}$ and $\underline{s}$ have non-empty presence-ranges;
ii) $\underline{s}$ is not unnegatable, i.e., its absence-range is non-empty;
and iii) the absence-range of $\underline{s}$ does not intersect the presence-range of $\underline{h}$.

And pervasion of $\underline{s}$ with $\underline{h}$ holds when:

i) the non-empty presence-ranges of $\underline{s}$ and $\underline{h}$ intersect;
and ii) if $\underline{h}$ is locatable in the absence-range of any $\underline{t}$ then $\underline{t} \neq \underline{s}$.

Following Strawson, we can represent the three interpretations of the A-relation and contrast them with non-deviation and pervasion as follows:

$\underline{xAy}$	(1st inter.)	$a\bar{b} = 0$
	(2nd inter.)	$a\bar{b} = 0 \cdot \underline{a} \neq 0.$
	(3rd inter.)	$a\bar{b} = 0 \cdot \underline{a} \neq 0. \ \underline{b} = 0.$
$\underline{hNs}$		$(\underline{h}+.\underline{s}- = 0) \cdot \underline{h}+ \neq 0 \cdot \underline{s}- \neq 0 \cdot \underline{s}+ \neq 0.$
$\underline{sVh}$		$(\underline{h}+.\underline{s}+ \neq 0) \cdot -(\underline{h}+.\underline{t}- \neq 0 \cdot \underline{t} = \underline{s}).$

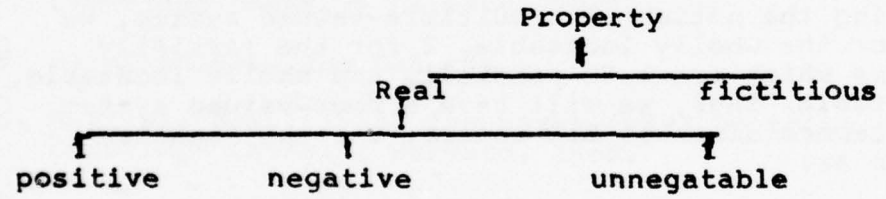
From above it is clear that the third interpretation of the A-relation is closer to the concept of non-deviation in Navya-nyāya except for the fact that the latter requires an additional condition. Navya-nyāya insistence on the non-emptiness of the presence-range or absence-range of properties pays dividend in the long run inasmuch as it makes the presupposition of a general statement (involving non-deviation or pervasion) clear. It should, however, be noted that both non-deviation and pervasion are much stricter relations compared to the A-relation.

Second, let us note that most inferences studied in Navya-nyāya try to locate a property (called sādhya 'inferable property' $\underline{s}$) in a particular locus (called pakṣa) with the help of another property (called hetu 'reason' $\underline{h}$). Thus, the predominant inference-pattern of Navya-nyāya corresponds to what W.V.Quine has called 'singular inference' (Quine, 1969, p.169). Hence, contrary to the belief of some modern interpreters of Indian logic, the Navya-nyāya

inference is not exactly a Barbara, but a singular inference. Chi has distinguished the standard Barbara from the singular inference by calling the latter Barbara-A and the former Barbara-B (p.13f).Navya-nyāya, however, allows inferences corresponding to Barbara-B, for it notes that pervasion relation is transitive (cf. tad-vyāpaka-vyāpakasya tad-vyāpakatvam, tad-vyāpya-vyāpyasya tad-vyāpyatvam).

Our last point will bring us closer to the explicit concern of this symposium, multiple-valued logic. The Navya-nyāya restrictions on negation, let me repeat, are instructive in more than one ways. To recapitulate briefly the Navya-nyāya position on negation: A property with an empty presence-range is called fictitious or unreal. We have called it unlocatable. Negation is viewed as an operation on real (non-fictitious) properties generating further real (i.e., locatable but negative) properties. Thus, a property with an empty absence-range is considered unnegatable in this system. For, although such a property is held to be real (since it is locatable), its negation would not generate a real (i.e., locatable) property.

It is possible to use some notions of multiple-valued logic under a special non-standard interpretation in order to represent the domain of properties in Navya-nyāya. Using "property" in the widest sense, we can construct the following tree to represent the branching of properties:



In ordinary three-valed system, such values as T, F and I are usually interpreted as "truth", "falsity" and "intermediate" (or, "undecided" or "neither true nor false"). Let us propose a completely different interpretation of values for the representation of the so-called real properties of Navya-nyāya. Our proposed three values are: P(for "positive"), N (for "negative") and U (for "unnegatable"). Now, we can have a standard three-valued negation as the following table will show:

$\underline{w}$	not-$\underline{w}$
P	N
N	P
U	U

This has the desirable outcome, viz.,

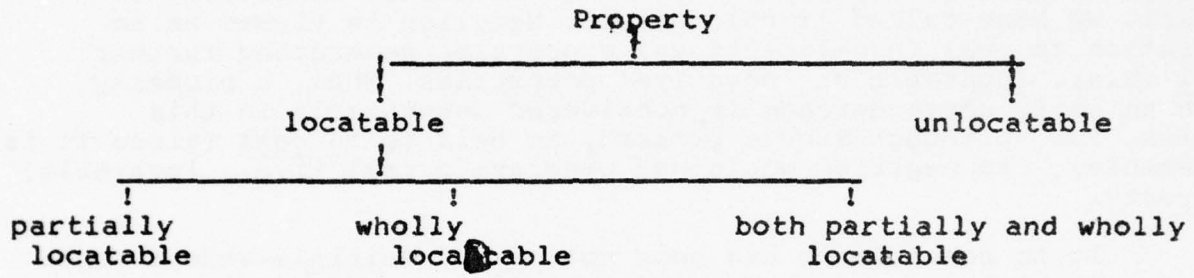
The presence-range of $\underline{w}$ = The absence-range of $\underline{\bar{w}}$.

The absence-range of $\underline{w}$ = The presence-range of $\underline{\bar{w}}$.

But by negating an unnegatable we get only another unnegatable (a

fictitious one). Further, since combination of an unnegatable with a positive yields, for Navya-nyāya, a positive property (and disjunction of a positive with an unnegatable yields an unnegatable), the corresponding tables for "AND" and "OR" can be constructed accordingly. But these tables will differ from the standard tables in some significant respects.

The problem of negation of the partially locatable properties can be tackled in another way. Let us construe the negation of a partially locatable property as both partially and wholly locatable. Then, we can agree with the following fourfold classification of properties:



We have seen, for example, that negation of chair-contact (a partially locatable property) yields a (negative) property which is both partially locatable (in some loci) and wholly locatable in other loci. Here, using the notion of a multiple-valued system, we can assign value 1 for the wholly locatable, 2 for the partially locatable, 3 for those which are both partially and wholly locatable, and 4 for the unlocatable. Thus, we will have a four-valued system with non-standard interpretation of all values, and the negation matrix can be written as:

w	not-w
1	1
2	3
3	2
4	4

Finally, we may note that despite the above oddities, the Navya-nyāya doctrine of negation is not very different from what is usually called "classical" or standard negation. The law of double negation, which roughly combines the law of contradiction and the law of excluded middle, is always satisfied by what Nyāya calls wholly locatable properties. (Only Raghunātha, a commentator of Gaṅgeśa, disputed this position, which I have mentioned above.) Thus, within the domain of wholly locatable properties, our standard notion of negation is not "mutilated" (to use Quine's phraseology).

Since the notion of partial location creates difficulty in interpreting negation in the standard fashion, Navya-nyāya recommends use of the technique of delimitors (cf. avacchedaka) by

which a partially locatable property can be parsed as a wholly locatable one so that negation can be given the desirable standard interpretation. And by declaring the unlocatables as unnegatable, Navya-nyāya saves another embarrassment that may possibily arise due to what is called "truth-value gaps" of such propositions as: "There is no gloden mountain" or "The son of a barren woman does not speak" (Udayana's example). Thus, inspite of the oddities encountered in Navya-nyāya theories, attempt has constantly been made here, with regard to negation, to follow what Quine has called the maxim of minimum mutilation.

REFERENCES

1. Aristotle. The Basic Works of Aristotle. Edited by Richard P. McKeon, New York, Random House, 1941.

2. Diṅnāga. Hetucakraḍamaru. See R.S.Y. Chi.

3. Gaṅgeśa. Tattvacintāmani. Edited with Mathurānātha's commentary by K. Tarkavagisha, Calcutta, Asiatic Society of Bengal, 1884-1901. Edited with Raghunātha's commentary and Gadādhara's sub-commentary, B. Bhattacharya and others, Beneras, Chowkhamba, 1913-1927.

4. Ingalls, D.H.H. Materials for the study of Navya-nyāya logic, Cambridge (Mass.), Harvard, 1951.

5. Mathurānātha. See Gaṅgeśa.

6. Matilal, B.K. (1968) The Navya-nyāya Doctrine of Negation, Cambridge (Mass.), Harvard, 1968.

7. -----. (1968a) "Gaṅgeśa on the concept of universal property (kevalānvayin)," Logic, Methodology and Philosophy of Science (edited by van Rootselaar and J.F.Staal, Amsterdam, North-Holland 1968.

8. ------. (1971) Epistemology, Logic and Grammar in Indian Philosophical Analysis, The Hague, Mouton & Co, 1971.

9. Quine, W.V. (1961) Methods of Logic, Revised Edition, New York, Holt, 1961.

10. -------. (1970) Philosophy of Logic, Englewood Cliffs, Prentice-Hall, 1970.

11. Raghunātha. See Gaṅgeśa.

12. Strawson, P.F. Introduction to Logical Theory, London, Methuen, (University Paperbacks), 1966.

13. Udayana. Nyāyakusumāñjali. Edited by L. Sastri Dravid. Benares, Chowkhamba, 1912.

14. Chi, Richard.S.Y. Buddhist Formal Logic, London, Luzac, 1969.

A Survey of Studies on Applications of Many-valued Logic in Japan

Tadahiro Kitahashi
Osaka University
Japan

1. Introduction

In Japan 2-valued logic and Boolean algebra was applied to logical design of relay circuits by A. Nakajima and M. Hanzawa in 1936 [1] as cited in the paper "The Development of Multiple-valued Logic as Related to Computer Science" by G. Epstein, G. Frieder and D. Rine.

The history of studies on applications of many-valued logic in Japan has experienced two heights of activities. The first one was observed on the period 1948-1955, where M. Goto [4], Y. Komamiya [8], and K. Yasu'ura [9] discussed relationships between many-valued logic and relay circuits. The second one spreaded from 1965 to 1974. Most of related papers to many-valued logic appeared on JIECE's in this period were concerned with multi-valued threshold functions. An actual and notable work in this height, however, was not their results but construction of a ternary arithmatic caluculator by T. Hasegawa and K. Shimada in 1970 [55].

Seminars on many-valued logic and its applications were held in Kyoto University in Feb. 19-21, 1970 [2] and July 14-16, 1971 [3] coodinated by T. Hasegawa.

2. Theoretical Approach

M. Goto applied 3-valued logic to analize and synthesize relay circuits for the first time in Japan. He introduced a different truth value from the conventional truth and false in order to represent transient states in switching of relays [4], [5], [6]. He also showed some types of solutions of many-valued logical equations for analysis and synthesis of relay circuits [7]. K. Yasu'ura proposed a method of realizing many-valued logical systems presented by M. Itoh [10] in relay circuits [9]. His method was based on transformation of one-wired multi-level signals to multi-wired two level signals.

Further, Y. Komamiya presented continuous logic and its applications [8]. H. Sugino and et al [11] proposed a method of detecting hazards in binary logical circuits using ternary logic.

M. Mukaidono [12], [13] and T. Takaoka [14] analized fail-safe circuits, assigning the failed state to a new truth value. Moreover, T. Tokaoka [15] and M. Miyata [16] extended the analysis to many-valued fail-safe circuits.

Asynchronous logical circuits also requires at least one truth value other than those needed for logic so as to indicate termimation of operations of circuits as well as asynchronous transmission schema [17]. T. Hasegawa [18] and Y. Koga [19] discussed synthesis of these circuits.

Such problems as mentioned above are those which necessarily requires many -valued logic for investigation.

At the same time, there appeared a number of papers presenting extentions of existing theories. These papers covered various areas.

First of all, multi-valued threshold logic should be mentioned. It was one of the most fertile themes which produced a lot of papers during the second height lasting from 1965 to 1974 in Japan. T. Kitahashi and et al described a necessary and sufficient condition of identifying ternary threshold functions, monotonic properties of them and a definition of Chow parameters of ternary functions based on their orthogonal expansions [20] ∿ [24]. They motivated studies on this theme in Japan, and some forty papers were published in this period.

H. Nomura showed that these properties can be extended to n(> 3)-valued threshold functions [25] ∿ [27]. He indicated eqivalence of a necessary and sufficient condition of identification of n-valued threshold functions and that of (n-1)-threshold threshold functions [25].

Detailed properties of many-valued, especially 3-valued, threshold functions were examined various points of view by T. Aibara [28] ∿ [30], Y. Takamatsu [31], [32], H. Mine and S. Fujita [33], [34], Y. Fujita [35], T. Haga [36], H. Mizuno [37], A. Imaizumi [38], H. Ataka [39], [40] and manyother researchers. It was noteworthy that Y. Fujita and et al proved the functional completeness of many -valued threshold functions [35], and T. Haga and et al pointed out a difference between many-valued logic and 2-valued logic; that is, in 2-valued logic assignment of truth values to any numerals preserve threshold functions, but in many -valued logic most of them fail to preserve threshold functions [36].

While Y. Fujita and et al also discussed synthesis multi-valued logical circuits by corresponding construction of feed forward circuits to composition of logical functions. Hence they mentioned classifications of logical functions and the simplest structures to realize given functions [41] ∿ [43].

A. Nozaki [44], [45] established a complete theory of composition of automata with many-valued inputs and outputs, which is an extention of the problems dealt with A. Minsky [46] and J. von Neumann [47].

M. Mizumoto and et al applied the fuzzy set theeory to the automata and formal language theory. They defined fuzzy automata and fuzzy languages and discussed their relationships to the well-known types of automata and formal languages [48].

It is possible to transmit signals with multi-level over low SN ratio transmission lines. Several multi-level codes were proposed and their band-widths and codewords interactions were examined [50], [51].

3. Hardware

H. Mine and T. Hasegawa and et al designed several logical and arithmetic circuits as well as systematic schemes. First, they constructed a pirot model of asynchronous transmission sheme based on the papers [52] ∿ [54]. Second, they constructed a system of ternary arithmetic operations installed in Kyoto University, based on investigations of forms of the system and circuits [55], [56].

Subsequent to their proposal, several ideas of ternary logical circuits were presented [57].

Acknowledgements

The author acknowledges with gratitude the continuous encouragement of Prof. Kokichi Tanaka. The author is also greatly indebted to Prof. Toshiharu Hasegawa in Kyoto University for his valuable suggestions and discussions; Prof. G. Epstein

Symposium Chairman, for his supports. Thanks is extended to Miss Taneyo Ukawa in the preparations of this paper.

References

[1] Nakajima, A. and M. Hanzawa, "The Theory of Equivalent Transformation of Simple Partial Paths in the Relay Circuits"(Part 1), (Part 2), Nippon Electrical Communication Engineering, No. 165 (Dec. 1936) and No. 167 (Feb. 1937).

[2] Hasegawa, T. (ed.), "1st Seminar on Many-Valued Logic and its Applications", Seminar Reports, Research Institute for Mathematical Science, Kyoto University No. 81, March 1970 (1st Semi. Rep. in Short)

[3] Hasegawa, T. (ed.), "2nd Seminar on Many-Valued Logic and its Applications", Seminar Reports, RIMS, Kyoto University No. 140, April 1972 (2nd Semi. Rep. in short)
(These reports are available in RIMS of Kyoto University.)

[4] Goto, M., "Applications of Ternary Logic to Relay Circuit Theory", Joint Conf. of Three Inst's. Related to Electrical Engn'r. in Tokyo, 1948.

[5] Goto, M., "Applications of Logical Equations to the Theory of Relay Circuits", Proc. IEEE, Vol. 69, 1948, p. 125.

[6] Goto, M., "Solutions of Logical Equations and its Applications", ETL Memorial Report, 1948, p. 1.

[7] Goto, M., "Some Types of General Solutions of Many-Valued Logical Equations with Many Variables", ETL Tech. Report, Vol. 20, No. 9, Sept. 1956, p. 671.

[8] Komamiya, Y., "Propositional Logic with Truth Values of Continuum Potencity and its Applications", ETL Tech. Report, Vol. 13, Dec. 1945, p. 498.

[9] Yasu'ura, K., "On Representations of Many-Valued Propositional Logic by Relay Circuits", Kyushu University. Tech. Report, Vol. 28, No. 5, 1955.

[10] Itoh, M., "On a Lattice of n-Valued Functions (n-valued Logic)", Kyushu Univ. Tech. Report, Vol. 28, No. 5, 1955.

[11] Sugino, H., Y. Inagaki and T. Fukumura, "A Method of Detection of Static Hazards Caused by Variation on Multi-Input Variables Using Ternary Logic", Proc. IECE, Vol. 50, No. 6, Jun. 1967, pp. 997-1105.

[12] Mukaidono, M., "Fail-Safe Logical Circuits Using Ternary Logic", 1st Semi. Rep. 1970, pp. 313-337.

[13] Mukaidono, M., "On the B-Ternary Logical Function — A Ternary Logic Considering Ambiguity — ", Trans. IECE, Vol. 55-D, No. 6, Jun. 1972, pp. 355-362.

[14] Takaoka, T., "On a System of Fail-Safe Logic" 1st Semi. Rep. 1970, pp. 290-312.

[15] Takaoka, T., "A Construction Method of Fail-Safe Systems for Many-Valued Logic", Trans. IECE, Vol. 54-C, No. 1, Jan. 1971, pp. 41-49.

[16] Miyata, M. and A. Fujioka, "Generalized Fail-Safe Sequential Machine", Trans. IECE, Vol. 54-C, No. 2, Feb. 1971, pp. 101-109.

[17] Mine, H., T. Hasegawa and Y. Koga, "Asynchronous Transmission Schemes for Digital Information", IEEE Trans. on Comm. Tech., Vol. COM-18, No. 5, Oct. 1970, pp. 562-568.

[18] Hasegawa, T., "On Asychronous Logical Circuits", 2nd Semi. Rep., pp. 395-416.

[19] Koga, Y., "On Synthesis of Asynchronous Logical Circuits Using Many-Valued Logic", 2nd. Semi. Rep., pp.379-394.

[20] Kitahashi, T. and et al, "On the Ternary Threshold Function", Memo, of Automaton and Information Research Group of IECE (IT 67-40), Nov. 1967.

[21] Kitahashi, T. and et al, "An Orthogonal Expansion of Multi-Valued Functions and its Application to the Realization with a Threshold Element", Trans. IECE, Vol. 52-C, No. 9, Sept. 1969, pp. 503-510.

[22] Kitahashi, T. and et al, "Characterizing Parameters of Ternary Logical Functions", Trans. IECE, Vol. 52-C, No. 10, Oct. 1969, pp. 641-648.

[23] Kitahashi, T. and et al, "Monotonicity of Ternary Logical Functions and its Applications to the Analysis of Ternary Threshold Functions", Trans. IECE, Vol. 53-C, No. 2, Feb. 1970, pp. 73-79.

[24] Kitahashi, T. and et al, "On the Complete Monotonicity as the Necessary and Sufficient Condition for a Ternary Logical Function to be a Threshold Function", Trans. IECE, Vol. 53-C, No. 8, Aug. 1970, pp. 507-513.

[25] Nomura, H., "Some Properties of Monotonicity on Many-Valued Logic and Their Applications to the Analysis of Many-Valued Threshold Functions", Trans. IECE, Vol. 55-D, No. 4, April 1972, pp. 245-252.

[26] Nomura, H., "Learning of the Multi-Threshold Function and its Application to the Synthesis of Logical Function", Trans. IECE, Vol. 55-D, No. 3, March 1972, pp. 194-201.

[27] Nomura, H., "Characteristic Vectors of Many-Valued Logical Functions and Their Applications to the Realization of Many-Valued Threshold Functions", Trans. IECE, Vol. 56-D, No. 1, Jan. 1973, pp. 17-24.

[28] Aibara, T. and Y. Takamatsu, "Complete Monotonicity for Multi-Valued Logic Functions", Trans. IECE, Vol. 53-C, No. 12, Dec. 1970, pp. 971-972.

[29] Aibara, T., "Tabulation of Characteristic Parameters of Ternary Three-Variable Threshold Functions", Trans. IECE, Vol. 54-C, No. 7, July 1971, pp. 606-611.

[30] Aibara, T. and Y. Takamatsu, "Complete Monotonicity for Multi-Valued Logic Functions and its Application to the Synthesis of Ternary Variable-Threshold Threshold Networks", Trans. IECE, Vol. 54-C, No. 12, Dec. 1971, pp. 1102-1109.

[31] Takamatsu, Y. and T. Aibara, "Realization of Ternary Threshold Functions Using a Map", Trans. IECE, Vol. 53-C, No. 4, April 1970, pp. 263-265.

[32] Takamatsu, Y., K. Murakami and T. Aibara, "Isobaricity for a Set of Multiple -Valued logical Functions and its Application to the Synthesis of Multithreshold Elements", Trans. IECE, Vol. 57-D, No. 9, Sept. 1974, pp. 534-541.

[33] Mine, H. and S. Fujita, "A Method for Testing and Realization of Three-Valued Threshold Functions", Trans. IECE, Vol. 54-C, No. 11, Nov. 1971, pp. 965-971.

[34] Mine, H. and S. Fujita, "Decomposition of Multi-Valued Switching Functions and its Application to Threhold Networks", Trans. IECE, Vol. 56-D, No. 12, Dec. 1973, pp. 691-725.

[35] Fujita, Y., T. Kitahashi, T. and K. Tanaka, "The Functional Completeness of Many-Valued Threhold Functions", Trans. IECE, Vol. 53-C, No. 5, May 1970, pp. 341-342.

[36] Haga, T., K. Abe and T. Fukumura, "A Problem in the Many-Valued Threshold Logic and its Solution", Trans. IECE, Vol. 55-D, No. 8, August 1972, pp. 514-522.

[37] Mizuno, H. and M. Kimura, "Testing and Realization of Ternary Threshold Functions", Trans. IECE, Vol. 55-D, No. 6, Jun. 1972, pp. 379-386.

[38] Imamiya, A., S. Noguchi and J. Oizumi, "Ternary Threshold Logic Networks Realizing Three-Valued Multithreshold Function", Trans. IECE, Vol. 56-D, No. 6, Jun. 1973, pp. 333-340.

[39] Ataka, H., "On the Definition of Ternary Threshold Functions", Trans. IECE, Vol. 54-C, No. 2, Feb, 1972, pp. 180-182.

[40] Ataka, H., "On the Unateness of Ternary Functions", Trans. IECE, Vol. 55-D, No. 6, Jun. 1972, pp. 398-399.

[41] Fujita, Y. and et al, "A Consideration on Compositions of Many-Valued Logical Functions", Trans. IECE, Vol. 55-A, No. 5, May 1972, p. 245.

[42] Fujita, Y. and et al, "An Extension of the Theorem of Isomorphism Concerning the Compositions of Many-Valued Switching Functions", Trans. IECE, Vol. 55-D, No. 6, Jun. 1972, p. 399.

[43] Fujita, Y. and et al, "Homomorphism of Logical Functions and Their Applications", Proc. Symposium on Theory and Applications to Logical Design, New York, May 1971, pp. 36-41.

[44] Nozaki, A., "Many-Valued Logic and Automata", 1st Semi. Rep., 1970, pp. 176-206.

[45] Nozaki, A., "Many-Valued Logic and Automata", 2nd Semi. Rep. 1972, pp. 50-60.

[46] Minsky, A., "Some Universal Elements for Finite Automata", Automata Studies, 1956.

[47] von Neumann, J., "Probabilistic Logic and the Synthesis", ibid.

[48] Mizumoto, M., J. Toyoda and K. Tanaka, "Some Considerations on Fuzzy Automata", J. Comp. & Syst. Sci., Vol. 3, No. 4, 1969, p. 409.

[49] Mizumoto, M., J. Toyoda and K. Tanaka, "General Formulation of Formal Grammars", Infor. Sci., Vol. 4, No. 1, 1972, p.87.

[50] Asabe, T., H. Nakanishi and Y. Tezuka, "On Cyclic Multilevel Codes", Trans. IECE, Vol. 55-A, No. 2, Feb. 1972, pp. 67-75.

[51] Asabe, T., H. Nakanishi and Y. Tezuka, "Spectral Density of Cyclic Multilevel Code", Trans. IECE, Vol. 55-A, No. 10, Oct. 1972, pp. 511-516.

[52] Hasegawa, T. and et al, "Tunnel Diode Circuits for Ternary Logic", Proc. IECE, Vol. 47, No. 10, Oct. 1964, pp. 58-64.

[53] Mine, H., T. Hasegawa and et al, "A Construction of Ternary Logic Circuits", Trans. IECE, Vol. 51-C, No. 12, Dec. 1968, pp. 573-580.

[54] Mine, H., T. Hasegawa and et al, "Construction and Analysis of a Tri-Stable Circuit and its Application to Ternary Feedback Shift Resister", Trans. IECE, Vol. 52-C, No. 8, August 1969, pp. 443-450.

[55] Mine, H., T. Hasegawa and R. Shimada, "A System of Ternary Arithmatic Operations", Proc. Information Processing Society of Japan, Vol. 12, No. 9, Sept. 1971, pp. 528-533.

[56] Mine, H. T. Hasegawa and R. Shimada, "Ternary Four Arithmatic Operations", Trans. IECE, Vol. 54-C, No. 1, Jan. 1971, pp. 66-73.

[57] Kitamura, Z., H. Terada and T. Takai, "Ternary Logic Circuits Using ESAKI Diodes", Trans. IECE, Vol. 53-C, No. 11, Nov. 1970, pp. 807-814.

A Critical Survey of Many-valued Logics 1966-1974

Robert G. Wolf
Southern Illinois University, Edwardsville
U. S. A.

1. Introduction

The basic material on which is based this overview of trends in the field of many-valued logic is included in a bibliography which was distributed during the sessions of the 1975 Symposium. The bibliography is too long to be included in the Proceedings but is available to anyone who contacts me at SIU. The present survey is meant both to introduce the bibliography and to draw out some of the patterns discernible in the field over the past ten years.

The bibliography follows the format laid down by Rescher /23/ for three basic reasons: 1) Rescher's bibliography is both relatively comprehensive and accurate. It would be a needless duplication of effort to redo his excellent work; 2) Rescher's bibliography is easily accessible to a wide audience, incorporated as it is in the only introductory level textbook in the field and hence his format should be faimilar; and 3) most importantly, Rescher's format with its triple listings is the most useable format for anyone actually working in the field.

The bibliography is divided into three sections. The first is a chronological listing of all the items with full bibliographical information, including information about reviews. The items are listed year by year and within each year, by the author's last name, with small case letters to differentiate items by the same author published in the same year (e.g., Georgescu 1971f). To facilitate use with the biliography in /23/, items from that bibliography are not reapeated except in those cases where additional bibliographical information is available. In those cases, Rescher's numbering is adopted.

While the main focus of the bibliography is for the period following 1965 (where the bibliography in /23/ ends), items discovered for the period 1910-1965 which are not in /23/ are listed and given numbers compatible with Rescher's. The existence of such items, given the comprehensiveness of Rescher's work, seems due to the narrower focus arising out of the introductory and philosophical orientation of Rescher's textbook. Items not strictly logical or philosophical tended to be excluded. This leaves the important areas of Post algebras and of computer applications, to name just two, untreated by Rescher. It is hoped that most of the important work in these fields is covered in the present bibliography.

The second section of the bibliography is an author listing with the titles and years of all items. The third section is a subject listing of all items with author and years provided. The subject headings which are used diverge slightly from those in /23/.

Even a quick glance at the bibliography (of approximately 800 items) shows that a tremendous amount of work has been done in the field over the last ten years and a comparison of the subject listing in /23/ with the current one indicates that research interest has shifted from that of previous decades. Our discussion of these shifts and of current emphases can be centered around three different areas: philosophy, logic and computer technology.

2. Philosophical Research

While the initial impulse towards many-valued logic (except in the case of Emil Post) seems to have been philosophical, the general trend in current work has been to move away from a concern with philosophical problems. Indeed most of the philosophical discussion in recent years has become increasingly critical of the philosophical applicability of many-valued logic and has been devoted to considering other possible approaches (though as we shall see, there are noteable exceptions). Discussion of future contingents and determinism has fallen mostly by the wayside (Seeskin's treatment in /30/ and /31/ is almost the only recent published discussion) as has discussion of liar-type paradoxes using many-valued logic (see however /10/ and /11/). Even discussion about the philosophical validity of other truth-values has almost disappeared; the only recent discussions of any substance seem to be those in chapter three of /23/ (which surprisingly has elicited almost no discussion) and in /46/.

The claims of many-valued logic as a useful tool for the explication of philosophical problems have been challenged either by modal and other intensional logics (see, e.g., /15/) or by van Fraassen's presuppositional logics, which avoid a third truth-value by allowing the values of molecular propositions to be undefined (see /39/, /40/ and /41/). Van Fraassen's approach has been applied to several philosophical problem areas; see /42/, /33/, /36/ and /17/. The superiority of this approach over that of positing an actual third truth-value has recently been challenged; see /34/ and /44/.

One area of philosophical concern in which a comprehensive attempt has been made to implement a many-valued logic approach is by Leonard Goddard and Richard Routley (and more recently by Ross T. Brady) in a series of papers culminating (for the time being) in the monmumental work by Goddard and Routley /9/. Goddard and Routley are concerned with interpreting the third value in a three-valued logic not as a truth value but as representing non-significance and applying the logics thus generated to philosophical arguments that claim that certain statements (e.g., those involving category mistakes) are without significance. Their book is perhaps the most important single work in the last ten years in the area of philosophical research into many-valued logic both for its detailed philosophical discussion and for the technical logical apparatus developed.

3. Logical Research

While philosophical research in many-valued logic has not noticeably expanded over the past ten years, interest among technical logicians has increased tremendously. While many logicians who work in this area have a definite philosophical orientation, many do not (e.g., Alan Rose) and even those who do do not limit their research to what is directly philosophically oriented (e.g., Peter W. Woodruff). Thus the work of the logician should not be confused with that of a philosopher. Overlaps may occur, but - in general - the logician is concerned with the formal properties of many-valued logics, whatever their applications may or may not be.

While work continues on the Hilbert-style axiomatic formulations of many-valued logics and their fragments (Alan Rose, whose works should be collected together and published in a format which would allow a wider recognition of his achievements, has been especially prolific in this area), the main focus of research has switched to three other areas. One area is that of the syntactic properties of many-valued logics, e.g., the definitions of connectives not possible in a two-valued logic, and the functional completeness of the logics with such connectives. A related area of more recent vintage is the question of the so-called 'structural completeness' of such logics. This research (done mostly by Polish logicians) is concerned with the

admissibility and derivability of rules of inference in various logics. (There are already far too many items in these areas for even a brief mention.)

Another area has been the construction of viable semantics for many-valued logics. To a great extent this involves the extension of the epochal work of Kripke in modal logic (see /2/, /3/, /6/ and /38/) and that of Gentzen in intuitionistic logic (see /16/, /24/, /25/, /27/, /28/ and /35/).

A third area (whose development has been spurred by its usefulness for computer applications) is the development of algebraic semantics for many-valued logics, especially the Post algebras and their generalizations. The investigation of Post algebras (and Łukasiewiczian algebras) shows perhaps the most striking growth of any area of purely logical concern and has taken on a life of its own, with work being done on such algebras as algebras and not as models for logics. The pioneering work of G. C. Moisil has recently been drawn together and published in a single volume /21/. More recent discussions can be found in /5/ and in later papers by Cignoli, while there exists a magisterial study of the entire field by Rasiowa /22/(which possesses the further advantage that it has been set up for use as a textbook).

Special mention should be made of /43/, which is relatively unknown and which, to my mind, presents a model of how a many-valued logic can be presented and developed. Woodruff presents for each logic he considers several Hilbert-style axiomatizations, Fitch-style natural deduction formulations, Gentzen-style consecution formulations (building upon unpublished work of Belnap reported in /20/) and Moisil-style algebraic formulations, proving appropriate completeness results and equivalence results among the various formulations. Woodruff also (using albegraic formulations) makes a significant generalization of the notion of "many-valuedness" of a logic, which allows him to formulate interesting systems of many-valued constructive logics (see also /45/), many-valued modal logics (see also /26/and /32/) and many-valued relevant logics (see /46/), none of which are in any straightforward sense truth-functional.

Work on many-valued logics has also been done by logicians with an eye to the light that they throw on modal, relevant and intuitionistic logics. The question of the finite model property for logics leads inevitably to a study of finite (or infinite but recursively-specifiable) matrices which will not be characteristic for such logics, but which do allow the proof of significant theorems. (See the basic work of Łos /18/ and the extended discussion in /37/). Work on the independence of various axioms for logics also involves the use of finite matrices; see, e.g., /1/ and /4/. Finitely many-valued logics arise also in the "neighborhood" of both modal and intuitionistic logics. The basic work of Scroggs /29/ showed that any normal extension of S5 would be a finitely many-valued logic. See /7/ for similar results in the relevance logic field and /19/ for a finite modal logic. Many of the intermediate logics are also finitely many-valued. For an early discussion of the intermediate logics, see /12/ and /13/; for a more recent survey, see /14/.

4. Computer Research

As spectacular as the growth of work in Post algbras has been, the growth of work done on the computer applications of many-valued logics has overshadowed it. It is no exaggeration to insist that such applications are the most important area in many-valued logics today and, strictly speaking, the only useful area of research. Even a brief discussion is beyond our capacity here; fortunately the present audience is in general most familiar with this area and given the excellent survey by Epstein and Frieder and Rine /8/, none is really needed. We will only mention that much of the work on the algebra of many-valued logics and on special connectives and functions in such logicss has been done with an eye towards computer applications. If present trends continue, the majority of work done on many-valued logics is likely to be

motivated by the consideration of computer applications.

5. Conclusion

We have divided the work in many-valued logics into three areas. To conclude we shall make some very tentative comments on the sort of work being done in various parts of the world. Philosophical discussion of many-valued logics has recently become the concern almost exclusively of the philosophers in English-spealing countries, noteably the United States. Logical discussion of such logics is more widely spread. Here too the majority of work seems to arise from America and the other English speaking countries. The Poles in recent years however have greatly increased their output in this field, concentrating on questions about the structural properties of logics, using the Tarskian notion of the logical-consequence relation as the basis of their research. The Japanese have also done an appreciable amount of work in this area. Russian work in the logical area seems to revolve mainly around questions about special connectives and functions and the completeness of such logics. In the sub-areas of Kripke-style semantics and Gentzen-style constructive "semantics", no clear trend seems discernible

In algebraic work the leaders seem to be the Soviet Union, Roumania and Poland, with increased American interest being shown in the past two or three years (sparked I believe by interest in computer applications). In the field of computer work, the United States and the Soviet Union have done the majority of the work, with Japan also doing a significant amount of work (all of which seems to reflect the importance of technology in those countries).

English and Russian are the languages in which the vast majority of research is reported, with English having the clear edge, partly (I feel) because it has been adopted by researchers in the Scandinavian countries, Poland and Japan as the normal medium for international reporting of their results. Even Roumanian scholars seem to be moving away from the use of French to the use of English as their chosen language of reporting their research.

Finally we might mention that almost no work in these areas seems to be done on the African continent, the Indian subcontinent and in China. Whether this is merely a reflection of the inadequacy of the reviewing and reporting of such work in the standard reviewing journals or whether it reflects some deeper couse I shall leave to others to decide.

Acknowledgements

The author wishes to acknowledge the help of J. Michael Dunn and George Epstein in locating items on many-valued logic which otherwise he would have missed. I am grateful to them. I also wish to thank the Graduate School of SIU, Dean Bailey of the School of Humanities and Professor Carol Keene, Chairperson of the Department of Philosophical Studies for their financial assistance in the typing and duplicating of the bibliography. Finally I would like to thank my wife, Susan, for putting up with me during the compilation and preparation of the bibliography.

References

/1/ Anderson, Alan Ross and Nuel D. Belnap, Jr., Entailment: The Logic of Relevance and Necessity. Volume I. Princeton (Princeton University Press) 1975.

/2/ Bertolini, Fernando, "Kripke Models and Manivalued Logics," Symposia Mathematica, Vol. V, London (Academic Press) 1970, pp. 113-131.

/3/ Bertolini, Fernando, "Kripke Models and Many-valued Logics. II," Rendiconti del Seminario Matematico dell'Università di Padova, vol. 44(1970), pp. 355-381.

/4/ Chidgey, John R., On Entailment. A Consideration of Some of the Formal Properties of the Anderson and Belnap System E of Entailment and of Related Systems. Ph. D. dissertation: Victoria University of Manchester, 1974.

/5/ Cignoli, Roberto, Moisil Algebras. Notas de Logica Matemática, N° 27, Bahia Blanca (Instituto de Matématica, Universidad Nacional) 1970. 47 pp.

/6/ Dahn, B., "Kripke-style Semantics for Some Many-valued Propositional Calculi," Bulletin de l'Académie Polonaise des Sciences, Série des Sciences Mathématiques, Astronomiques et Physiques, vol. 22(1974), pp. 99-102.

/7/ Dunn, J. Michael, "Algebraic Completness Results for R-mingle and Its Extensions," The Journal of Symbolic Logic, vol. 35(1970), pp. 1-13.

/8/ Epstein, George, Gideon Frieder and David C. Rine, "The Development of Multiple-valued Logic as Related to Computer Science," Computer, vol. 7, no. 9(1974), pp. 20-32.

/9/ Goddard, Leonard and Richard Routley, The Logic of Significance and Context. Volume I. Edinburgh (Scottish Academic Press) 1973 and New York (Halsted Press) 1973.

/10/ Herzberger, Hans G., "Truth and Modality in Semantically Closed Languages," The Paradox of the Liar, edited by Robert L. Martin, New Haven (Yale University Press) 1970, pp. 25-46.

/11/ Herzberger, Hans G., "Dimensions of Truth," Journal of Philosophical Logic, vol. 2(1973), pp. 535-556.

/12/ Hosoi, Tsutomu, "On Intermediate Logics, I," Journal of the Faculty of Science, University of Tokyo, section I, vol. 14(1967), pp. 293-312.

/13/ Hosoi, Tsutomu, "On Inter mediate Logics, II," Journal of the Faculty of Science, University of Tokyo, section I, vol. 16(1969), pp. 1-12.

/14/ Hosoi, Tsutomu, and Katuzi Ono, "Intermediate Propositional Logics (A Survey), Journal of Tsuda College, forthcoming.

/15/ Kamp, Hans, "Two Theories About Adjectives," forthcoming.

/16/ Kirin, Vladimir G., "Gentzen's Method for the Many-valued Propositional Cacluli," Zeitschrift für mathematische Logik und Grundlagen der Mathematik, vol. 12(1966), pp. 317-332.

/17/ Lambert, Karel, "Logic and Microphysics," The Logical Way of Doing Things, edited by Karel Lambert, New Haven (Yale University Press) 1969, pp. 93-117.

/18/ Łos, Jerzy, O matrycach logicnych. Wrocław (Prace Wrocławskiego Towarzystwa, Naukowego, series B, no. 19) 1949. English translation ("On Logical Matrices") by Dolph E. Ulrich in /37/ pp. 71-129.

/19/ McCall, Storrs, and Arnold Vander Nat, "The System S9," Philosophical Logic, edited by J. W. Davis, D. J. Hockney and W. K. Wilson, Dordrecht (D. Reidel Publishing

Company) 1969, pp. 194-214.

/20/ Meyer, Robert Kenneth, Topics in Modal and Many-valued Logic. Ph. D. dissertation: University of Pittsburgh, 1966. Ann Arbor (University Microfilms) 1966.

/21/ Moisil, Grigor Constantin, Essais sur le Logiques non-chrysippiènnes. Bucharest (Éditions de l'Académie de la République Socialiste de Roumanie) 1972.

/22/ Rasiowa, Helena, An Algebraic Approach to Non-classical Logics. Amsterdam (North-Holland Publishing Company) 1974.

/23/ Rescher, Nicholas, Many-valued Logic. New York (McGraw-Hill Book Company, Inc.) 1969.

/24/ Rousseau, George H., "Sequents in Many-valued Logic I," Fundamenta Mathematicae, vol. 61(1967), pp. 22-33.

/25/ Rousseau, George H., "Sequents in Many-valued Logic II," Fundamenta Mathematicae, vol. 67(1970), pp. 133-145.

/26/ Ruzsa, I., "Prior-type Modal Logics I," Periodica Mathematica Hungarica, vol. 4(1973), pp. 51-69.

/27/ Saloni, Zygmunt, "Gentzen Rules for the m-valued Logic," Bulletin de l'Académie Polonaise des Sciences, Série des Sciences Mathématiques, Astronomiques et Physiques, vol. 20(1972), pp. 819-826.

/28/ Saloni, Zygmunt, "The Sequent Gentzen System for m-valued Logic," Bulletin of the Section of Logic, Polish Academy of Sciences, Insistute of Philosophy and Sociology, Wrocław, vol. 2(1973), pp. 30-37.

/29/ Scroggs, Schiller Joe, "Extensions of the Lewis System S5," The Journal of Symbolic Logic, vol 16(1951), pp. 112-120.

/30/ Seeskin, Kenneth R., "Many-valued Logic and Future Contingencies," Logique et Analyse, n.s., vol. 14(1971), pp. 759-773; vol. 15(1972), p. 665.

/31/ Seeskin, Kenneth R., "Some Remarks on Truth and Bivalence," Logique et Analyse, n.s., vol. 17(1974), pp. 101-109.

/32/ Segerber, Krister, "Some Modal Logics Based on a Three-valued Logic," Theoria, vol. 33(1967), pp. 53-71.

/33/ Skyrms, Brian T., "Return of the Liar: Three-valued Logic and the Concept of Truth," American Philosophical Quarterly, vol. 7(1970), pp. 153-161.

/34/ Stiver, James LeRoy, Presupposition, Implication and Necessitation: A Critique of and Alternative to van Fraassen's Presuppositional Logic. Ph. D. dissertation: University of North Carolina at Chapel Hill) 1972. Ann Arbor (University Microfilms) 1973.

/35/ Surma , Stanisław J., "A Method of the Construction of Finite Łukasiewiczian Algebras and Its Application to a Getnzen-stile Characterization of Finite Logics," Report on Mathematical Logic, no. 2(1974), pp. 49-54.

/36/ Thomason, Richmond H., "Indeterminist Time and Truth-value Gaps," Theoria, vol. 36(1970), pp. 264-281.

/37/ Ulrich, Dolph Edward, Matrices for Sentential Calculi. Ph. D. dissertation: Wayne State University, 1967. Ann Arbor (University Microfilms) 1971.

/38/ Urquhart, Alasdair I. F., "An Interpretation of Many-valued Logic," Zeitschrift für mathematische Logik und Grundlagen der Mathematik, vol. 19(1973), pp. 111-114.

/39/ van Fraassen, Bas C., "Singular Terms, Truth-value Gaps and Free Logic," The Journal of Philosophy, vol. 63(1966), pp. 481-495.

/40/ van Fraassen, Bas C., "Presupposition, Implication, and Self-reference," The Journal of Philosophy, vol. 65(1968), pp. 136-152.

/41/ van Fraassen, Bas C., "Presuppositions, Supervaluations, and Free Logic," The Logical Way of Doing Things, edited by Karel Lambert, New Haven (Yale University Press), 1969, pp. 67-91.

/42/ van Fraassen, Bas C., "Truth and Paradoxical Consequences," The Paradox of the Liar, edited by Robert L. Martin, New Haven (Yale University Press) 1970, pp. 59-66.

/43/ Woodruff, Peter Worthing, Foundations of Three-valued Logic. Ph. D. dissertation: University of Pittsburgh, 1969. Ann Arbor (University Microfilms) 1969.

/44/ Woodruff, Peter W., "Logic and Truth-value Gaps," Philosophical Problems in Logic: Some Recent Developments, edited by Karel Lambert, Dordrecht (D. Reidel Publishing Company) 1970, pp. 121-142.

/45/ Woodruff, Peter W., "On Constructive Nonsense Logic," Modality, Morality and Other Problems of Sense and Nonsense. Essays Dedicated to Sören Halldén, Lund (CWK Gleerup Bokförlag) 1973, pp. 192-205.

/46/ Woodruff,Peter W., "Relevance and Many-valued Logic," Proceedings of the International Conference on Relevance Logics. In Memoriam: Alan Ross Anderson, edited by Kenneth W. Collier and Robert G. Wolf, forthcoming.

AD-A045 757 INDIANA UNIV BLOOMINGTON DEPT OF COMPUTER SCIENCE F/G 9/2
PROCEEDINGS OF THE 1975 INTERNATIONAL SYMPOSIUM ON MULTIPLE-VAL--ETC(U)
MAY 75 G EPSTEIN N00014-75-C-0449
UNCLASSIFIED MVL-75-001 NL

6 OF 6
AD A045757



END
DATE FILMED
11-77
DDC

Editorial Notes

1. The final versions incorporating all the authors' revisions for each of Drs. Grigolia's, Pinkava's, and Wójcicki's papers (p. 234, p. 20, and p. 240, respectively) were not received by us in time for these Proceedings, due to overseas mailing irregularities.

2. During our final preparations for publication, Dr. Michalski's abstract for his paper (p. 76) was inadvertently omitted from the text. His abstract for the first page of his paper, "Synthesis of Optimal and Quasi-Optimal Variable-Valued Logic Formulas", is as follows:

> This paper describes some ideas and algorithms for the synthesis of optimal and quasi-optimal variable-valued logic expressions. The central concept underlying the synthesis algorithm is that of a cartesian cover of one set against another set. The synthesis of covers is based on the algorithm A^q which employs the principle of disjoint stars.

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R & D (Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)	
1. ORIGINATING ACTIVITY (Corporate author) ~~Indiana University Foundation~~ Indiana University Bloomington, Indiana 47401	2a. REPORT SECURITY CLASSIFICATION-UNCLASSIFIED 2b. GROUP
3. REPORT TITLE 1975 International Symposium on Multiple-Valued Logic	
4. DESCRIPTIVE NOTES (Type of report and inclusive dates) Final Report: PROCEEDINGS OF THE 1975 INTERNATIONAL SYMPOSIUM ON MULTIPLE-VALUED LOGIC held	
5. AUTHOR (First name, middle initial, ~~last name~~) George Epstein Computer Science Department, Indiana University, Bloomington, Indiana	
6. REPORT DATE May 1975	7a TOTAL NO. OF PAGES 475 May 13-16, 1975
8a. CONTRACT OR GRANT NO. N00014-75-C-0449 b. PROJECT NO. NR-048-627/12-17-74 new c. d.	9a. ~~ORIGINATOR'S~~ REPORT NUMBER(S) MVL-75 001 9b. OTHER REPORT NO(S)(Any other numbers that may be assigned this report)
10. DISTRIBUTION STATEMENT Distribution of this Document is Unlimited	
11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Office of Naval Research
13. ABSTRACT PROCEEDINGS of the 1975 International Symposium on Multiple-Valued Logic	

UNCLASSIFIED

Security Classification

1 404 270

