VECTOR-MATRIX REPRESENTATION OF BOOLEAN ALGEBRAS AND APPLICATION TO EXTENDED PREDICATE LOGIC (EPL)—Part I

G. N. RAMACHANDRAN*

CSIR Distinguished Scientist, 'Gita', 10A Main Road, Malleswaram West, Bangalore 560 055, India.

ABSTRACT

This article deals with the application of the general $m \times n$ Boolean vector-matrix representation of the Theory of Relations to Boolean algebra BA-n, with $n \times n$ matrices. A new type of relation termed "reverse relation" is defined and it is found vitally important for both Boolean algebra (BA) and logic. In BA, it leads to a new type of non-closure—namely BA-m iteratively leading to a higher BA-n $(n > m)$ on reversal. BA-1, associated with propositional calculus (PC), necessarily leads to BA-2 for its full representation, the extended PC which we call as SNS (from syād (Sanskrit) = doubt). This doubtful state is a new state found to be essential for the completeness of PC. BA-3 is shown to be isomorphic to quantified predicate logic (QPL), provided the algebra of connectives is written in terms of what we have designated "canonical states", representable by BA-3 vectors. The algebra of the connectives has 64 "and" ($A(i,j)$, $i, j = 1$ to 8) and 64 "or" ($O(i,j)$) connectives, and a consistent, complete representation of QPL in terms of Boolean vectors, and 3×3 matrices has been worked out. Here again a new basic state "some" (Σ) is found to be essential, in addition to "all" ($\forall$) and "none" (Φ). This state which is there in Ancient Indian Logic of two thousand years ago, makes Godel's second (incompleteness) theorem for QPL understandable from a simple approach.

1. INTRODUCTION

THIS article, is in a way, the continuation of the previous article[1] published in this journal. In that the application of Boolean algebras BA-1 and BA-2 to propositional calculus or sentential logic was considered. By trial and error methods, we discovered that the next higher order Boolean algebra (BA-3) is a very suitable one for symbolizing quantified predicate logic and the various connective operators and states that occur in it. When this was done, just as BA-2 led to two new states, D (doubtful) and X (impossible) in addition to T and F, in propositional calculus, a new state, symbolised by Σ (some) was found to be necessary to get a complete set of all states in predicate logic, in addition to the usual $\forall$, $\exists$, and their negations. This, however, led to eight possible states by completing the associated BA-3, with four new additions to the standard four that are used in QPL, namely "for all" ($\check{\forall}$), "for none" (Φ), "there exists" ($\exists$) and "not for all" (Λ).

Although the requirements of predicate logic could be covered by using these eight states and connective operators (both matrix and non-matrix) as in SNS, it was felt that the full power of the matrix representation can be brought out only by considering a general n-valued logic which can be representated by BA-n. Even further, this BA-n and the matrix operators occurring for it turn out to be still further generalizable by using rectangular matrices of the type $m \times n$ in the theory of relations, connecting m different objects of one type with n different objects of another type. Therefore, this general theory of relations will be considered first. Then its reduction to a system with only $n \times n$ matrices become straightforward. Out of these, the particular cases of BA-1, BA-2 and BA-3 will be discussed, from the general standpoint, and their consequences to logic will then be described. In particular, the applications of BA-3 to quantified predicate logic turns out to be very novel and these will be described in some detail.

2. THEORY OF RELATIONS AND BOOLEAN MATRICES

To keep the tenor of this article at an elementary level, we shall illustrate the method of applying Boolean vectors and matrices for the Theory of Relations by means of a simple example, although general proofs of the statements we make can be readily formulated. We take two sets,—one designated $P(= p_1, p_2, p_3, p_4)$ consisting of 4 parents and the other designated $C(= c_1, c_2, c_3, c_4, c_5)$ consisting of 5 children. The forward relation from parent to child, which we may denote by C (standing for "child of") is represen-

* The author was formerly Professor of Mathematical Philosophy, Indian Institute of Science, Bangalore 560 012.

† This is Part I which will be followed by Part II in the next issue.

table by a "truth table", as in Table 1, in which, an entry 1 or 0 as C_{ij} means that the relation exists, or is absent between c_i and p_j. Thus, c_3 is the child of p_1 and p_2, while c_1 and c_2 are children of p_3 and c_4 of p_4; c_5 is not the child of any of p_1 to p_4, although it is included in the set $C \equiv \{c_j\}$. Similarly, for the relation "parent of", **P**.

TABLE 1

Truth tables for parent-child relationships

(a) **C** (Child of)

•	c_1	c_2	c_3	c_4	c_5
p_1	0	0	1	0	0
p_2	0	0	1	0	0
p_3	1	1	0	0	0
p_4	0	0	0	1	0

(b) **P** (Parent of)

	p_1	p_2	p_3	p_4
c_1	0	0	1	0
c_2	0	0	1	0
c_3	1	1	0	0
c_4	0	0	0	1
c_5	0	0	0	0

The relations in Tables 1(a) and (b) can be written as Equations (1a) and (1b) below. In this, the row vectors $<p|$ and $<c|$ stand for the Boolean vector representation of which constituents are present. Thus, the state vector (1 0 1 1) for $<p|$ indicates that p_1, p_3, p_4 are present and p_2 is absent. Then, the relational matrices in (1a) and (1b) are exactly as given in the two tables, namely 5×4 for $|C|$ and 4×5 for $|P|$. Thus, we have

$$<p|C| = <c|, \text{ for Table 1(a)} \tag{1a}$$

$$<c|P| = <p|, \text{ for Table 1(b)} \tag{1b}$$

It is obvious that the matrices $|C|$ and $|P|$ are transposes of one another, and they represent the two relations in (1a) and (1b), which are termed "reverses" of each other. The relation **a R b** read in the "reverse" direction as **b R′ a** has its matrix $|R'| = |R^t|$, the superscript 't' standing for "transpose".

The notation of a row vector as a "bra" vector ($<v|$), a column vector as a "ket" vector ($|v>$) and the relational matrix enclosed by two vertical lines (as in $|Z|$), follows the Dirac bracket notation in quantum mechanics (see [1], for fuller details). We write all equations from left to right, as this is the order in which logical relations are expressed—as in "**a** implies b" ($a \Longrightarrow b$), which has the notation **a I** = b in our nomenclature (see [1]), with $<a|$ and $<b|$ as 2-element Boolean "bra" vectors and $|I|$ as a 2×2 Boolean matrix, giving $<a|I| = <b|$.

We shall discuss two practical uses of relational matrices that are relevant to logic and Boolean algebra.

(a) Unary relation and its reverse:

If $<p'|$ is the Boolean vector representing a (partial) ensemble of the full set $\{p_i\}$, *eg.* (1 0 0 1), we may ask the question, "which c_j's are present, among the children, of the two parents p_1 and p_4 of this p′-ensemble?" The answer is given as the Boolean vector $<c'|$ in Eqn. (2a), which, for $<p'| = (1\ 0\ 0\ 1)$ yields $<c'| = (0\ 0\ 1\ 1\ 0)$ as in (2b)—namely, only the two children c_3 and c_4.

$$<p'|C| = <c'| \tag{2a}$$

$$<p'| = (1\ 0\ 0\ 1) \longmapsto <c'| = (0\ 0\ 1\ 1\ 0) \tag{2b}$$

Note that all the additions and multiplications involved in the vector-matrix product are as per standard Boolean algebra BA-1 (see [ref 1] for more examples), where the matrix $|C|$ is explicitly as shown below in (3a). The meaning and use of the matrix $|C^c|$ in (3b) will be clear in section 2(b)

$$|C| = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{3a}$$

$$|C^c| = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \end{bmatrix} \tag{3b}$$

In the same problem, if the question is reversed, with $<c'|$ as input, we obtain Eqn. (4a), yielding the vector $<p''|$ is given by (4b).

$$<c'|P| = <p''| \ (4a); \quad <p''| = (1\ 1\ 0\ 1) \tag{4b}$$

The interesting feature of this reversal of the relation **C** to the reverse relation **P** is that $<p''|$ in (4b) is not the same as $<p'|$ in (2b). The reason, in this particular case, is that c_3 is the son of p_1 but both the parents p_1 and p_2 are present in the set $\{p_i\}$. One parent (p_1) gives the child (c_3) in the forward direction using the relation **C**, but in the reverse direction, the relation **P** applied to the same child (c_3) gives *both* the parent p_1

and p_2—in general the maximal ensemble of parents possible. Thus, the 'reverse' relation does not have the nature of 'inverse' in ordinary algebra and the matrix $|Z|$ goes over into its *transpose* $|Z^t|$, on reversing the relation. This property of *transpose representing reversal of a relation* is vitally important for our applications of Boolean vectors and matrices to logic.

(*b*) *Dirac (matrix) product*:

Suppose, in Table 1a, we take the first of the above two examples and work out the Dirac product $\langle p'|P|c'\rangle$, as defined in [1], which is repeated as (5) below for ready reference

$$\langle a|Z|b\rangle = \sum_i \sum_j a_i Z_{ij} b_j = k_\alpha, \text{ a scalar} \tag{5}$$

In this, the value of k_α can only be 0 or 1 in Boolean algebra (the need for the subscript α in k_α will be clear in Eqn. (7) below). If $k_\alpha = 1$, it means that the answer to the question "Is there at least one a_i and one b_j in the ensembles a and b, that are related by the relation **Z**?" is "yes". Similarly, $k_\alpha = 0$ means that the relation **Z** does not connect any a_i with any b_j in the two ensembles represented by $\langle a|$ and $\langle b|$, that are provided.

In the same way, we may form a Dirac product using the complement of $|Z|$, namely $|Z^c|$, where

$$Z_{ij}^c = 1 - Z_{ij}, \text{ its complement in BA-1} \tag{6}$$

The matrix $|Z^c|$ represents the "non-relation" of **Z**, so that if there is 1 in position (i, j) in $|Z^c|$, then a_i and b_j are *non-related* by the relation **Z**. We thus have an equation similar to (5) for k_β, namely

$$\langle a|Z^c|b\rangle = \sum_i \sum_j a_i Z_{ij}^c b_j = k_\beta \tag{7}$$

and the properties of k_β for $\mathbf{Z}^c$ are the same as those of k_α for **Z**.

(*c*) *Binary relation expressed via Dirac products*:

We now consider the truth value of the binary relation a**Z**b, expressed *via* the state vectors $\langle a|$, $\langle b|$, and the matrix $|Z|$ representing the relation **Z**, and its derived complement $|Z^c|$. The truth value of the relation is expressible in terms of *two* Boolean scalars k_α and k_β, which form a Boolean 2-vector $(k_\alpha\, k_\beta)$. In order to see the various possibilities, three examples can be taken, as in Sl. Nos 1, 2, 3 of Table 2.

Thus, the non-relation ($\mathbf{C}^c$) does not exist for the Row 1, and the relation (**C**) exists, while the opposite is the case of Row 2 of Table 2. We shall say that the truth value is "true" (T) for the former, and "false" (F) for the latter, which agrees with the idea that only one of the two—"relation **C**" or "non-relation $\mathbf{C}^c$" exists for these cases. We represent these by the Boolean vectors (1 0) and (0 1) respectively, as in SNS[1].

However, it is possible that both **C** and $\mathbf{C}^c$ are not absent (Row 3 of Table 2). This gives $(c_\alpha\, c_\beta) = (1\ 1)$, when state of p**C**c is "indefinite" (D, standing for "doubtful"). Row 4 shows one more example, namely of *both* the relations **C** and $\mathbf{C}^c$ being non-existent. It is easy to show that such a situation will occur only if either the vector $\langle p'|$ or $\langle c'|$ is zero for all its components, when a relation between p and q is "impossible"—represented by (0 0). The letter X indicates this state.

(*d*) *Summary of Section 2*:

Thus, the truth value of a binary relation requires a BA-2 representation with *two* scalars in the form of Dirac products so that the relation a**C**b = c becomes representable as (8a) and (8b):

$$\mathrm{a\,C\,b} \Longleftrightarrow (a|C|b) = (c_\alpha\, c_\beta) = \mathbf{c} \text{ (say)} \tag{8a}$$

where c_α and c_β are given by Dirac products:

$$\langle a|C|b\rangle = c_\alpha;\quad \langle a|C^c|b\rangle = c_\beta \tag{8b}$$

TABLE 2

Dirac product values for four typical examples of the relation **C**.

Sl. No.	$\langle p'\|$	$\langle c'\|$	k_α	k_β	Logical state $\mathrm{k} = (k_\alpha\, k_\beta)$ of the relation **C**
1.	(0 0 1 0)	(1 1 0 0 0)	1	0	T
2.	(0 0 1 1)	(0 0 1 0 1)	0	1	F
3.	(1 0 0 1)	(0 0 1 1 0)	1	1	D
4.	(0 0 0 0)	(any values)·	0	0	X

We shall use $(a \mid C \mid b)$ to indicate the SNS state $(c_\alpha c_\beta)$ of $\mathbf{a C b = c}$. Also, since the matrix $\mid C^c \mid^c = \mid C \mid$ itself, the bracket giving the truth value for the relation $\mathbf{a C^c b}$ is

$$(a \mid C^c \mid b) = (c_\beta\, c_\alpha) = \mathrm{d} = \mathrm{c\,N} \tag{8c}$$

where c_α and c_β are the same as in (8b) (see [1] for the definition of SNS negation N).

Obviously, the matrix $\mid C \mid$ of (8a) can be written as

$$\mid C \mid = \sum_{\lambda} \sum_{\mu} \mid C_{\lambda\mu} \mid$$

such that each $\mid C_{\lambda\mu} \mid$ *has only one non-zero element* at location (λ, μ) in the matrix. Such a matrix $\mid C_{\lambda\mu} \mid$, with a single 1 in it, may be called a "singular matrix". In terms of these, we have

$$(a \mid C \mid b) = \sum_{\lambda} \sum_{\mu} (a \mid C_{\lambda\mu} \mid b) \tag{8d}$$

The use of such singular matrices in quantified predicate logic is to be found in section 6.

3. Types of relations in Boolean Algebra

(a) *Reversal of a relation, in general:*

The above general treatment of a relation Z expressible by an $m \times n$ Boolean matrix, can be seen to be fully acceptable within the range of the mathematics of set theory, Boolean algebra and matrix theory. However, when the above formulae are applied to logic, they lead to extremely interesting results. Some of them turn out to be quite novel, not only in leading to simplified procedures in the mathematical treatment of the theory of logic, but even to some new concepts in logic itself.

The concept of the "reversal" of relations does not appear to be generally recognized in the literature. We saw one aspect of it connected with unary relations earlier in subsection 2(i), Eqs (2b) and (4b). Thus the ensemble $\langle p' \mid = (1\ 0\ 0\ 1)$ of parents gave the ensemble $\langle c' \mid = (0\ 0\ 1\ 1\ 0)$ for children; but when the operation was reversed, we obtained $\langle p'' \mid$ as $(1\ 1\ 0\ 1)$, different from $\langle p' \mid$. It can be shown that $\langle p'' \mid$ is really the set of "all possible" parents of the children $\langle c' \mid$, and that the members of $\langle p' \mid$ are *included* in $\langle p'' \mid$. This idea that a Boolean vector-matrix operation gives the maximal set of members contained in the vector generated by it, is a very important one for our discussion. The ensemble of elements like $\langle p' \mid$ and $\langle c' \mid$ form a 'lattice' of abstract algebra. In fact Boolean algebras form "fully complemented distributive lattices" (See [2]).

(b) *Reversal of binary relations:*

We now consider the technique of reversing a binary relation. We start with 8(a, b, c). For this, we suppose that the state of the vector $\mathrm{c} = (c_\alpha c_\beta)$ is given and so also is the relation Z. We are then required to find out the state that can be deduced for $\langle b \mid$ given that of $\langle a \mid$, or *vice versa*. Taking the former first, the relation between a and b can be given, in general, in *terms of Boolean vector-matrix products involving* the matrices $\mid C \mid$ and $\mid C^c \mid$, as in Eqns (9a to b):

$$\mathrm{c = T} \text{ yields } \langle a \mid C \mid \otimes (\langle a \mid C^c \mid)^c = \langle b \mid \tag{9a}$$

$$\mathrm{c = F} \text{ yields } \langle a \mid C^c \mid \otimes (\langle a \mid C \mid)^c = \langle b \mid \tag{9b}$$

$$\mathrm{c = D} \text{ yields } \langle a \mid C \mid \oplus \langle a \mid C^c \mid = \langle \mathit{1} \mid \tag{9c}$$

$$\mathrm{c = X} \text{ yields } \langle b \mid = \langle \mathit{0} \mid, \text{ and } \langle a \mid = \langle \mathit{0} \mid \tag{9d}$$

In (9c, d), $\langle \mathit{1} \mid$ stands for a state vector with 1 for all components, and $\langle \mathit{0} \mid$ for one with 0 for all the components.

In these equations, it is important to remember that, in general, the vectors $\langle a \mid Z \mid = \langle d \mid$ and $\langle a \mid Z^c \mid = \langle e \mid$ are not complementary to one another and $\langle d \mid e \rangle$ need not be a null vector. Secondly, the vector $\langle b \mid$ yielded by the l.h.s's of (9a), or (9b) is what may be called the "maximal" vector, containing all possible elements b_i which can be non-zero. In an actual case, the vector $\langle b' \mid$ can have 1's for any, or all, of the b_i's in the maximal vector $\langle b \mid$; but not all b_i's can be zero (which will correspond to the case of $\mathrm{c = X}$ of (9d)). In this sense, the vital factor in each of (9a) and (9b) is the second of the two terms joined by the Boolean product and the first term only selects out of these, those that go into the vector $\langle b \mid$.

On the other hand, in (9c) the result for $\langle b \mid$ is $\langle \mathit{1} \mid$, which stands for the vector containing unities for all the elements $b_i (i = 1 \text{ to } n)$ of the vector $\langle b \mid$— namely $(1\ 1\ 1 \ldots 1)$, corresponding to the full set. In the same way, if $\mathrm{c = X}$ as in (9d), the vector $\langle b \mid$ is $\langle \mathit{0} \mid$, which, indicates that the set B represented by $\langle b \mid$ is a null set.

(c) *Boolean addition and multiplication of vectors:*

Since the n generators of BA-n are non-intersecting and one is not included in the other, the corresponding vectors form a basic set $\langle a_i \mid$, $i = 1$ to n, for the algebra. In the standard formulation of Boolean algebra[2], as a special type of lattice, the only relations through which these can combine are *via* the operations of Boolean addition $\oplus$ and Boolean multiplication $\otimes$ (both of which are distributive with respect to one another), leading to the equations

$$\langle a| \oplus \langle b| = \langle c| \Longleftrightarrow a_i \oplus b_i = c_i, \quad i = 1 \text{ to } n \tag{10}$$

$$\langle a| \otimes \langle b| = \langle d| \Longleftrightarrow a_i \otimes b_i = d_i, \quad i = 1 \text{ to } n \tag{11}$$

All the axioms of Boolean algebra are satisfied by the n-vector representation, if it is noted that complement $\langle a^c|$ of $\langle a|$ can be defined by

$$\langle a^c| = (a_1^c, \ldots a_i^c \ldots a_n^c), \tag{12a}$$

$$\text{where} \quad a_i^c = 1 - a_i \tag{12b}$$

and the set of all 2^n vectors of BA-n is closed with respect to the application of the operations in (10), (11), and (12).

As regards matrix multiplications of the type we have envisaged in Section 2 (namely by $n \times n$ matrices in BA-n), they employ only the types of elementary operations characteristic of BA-1, *via* $\oplus$ and $\otimes$, applied to 1 and 0 of this algebra. Hence, the set of 2^n vectors of BA-n is closed with respect to the operations of these Boolean matrix multiplications also. However, with respect to its application to logic, Boolean matrix operators and the symbolic Boolean operators of BA's, have entirely different interpretations, as will be shown below (See Sections 6(d), (e), (f)).

(*d*) *Closure of Boolean algebras*:

The reversal of the binary relation $(a \mid Z \mid b) = c = (c_\alpha\ c_\beta)$, may be symbolically represented by

$$(c \mid \overleftarrow{Z} \mid a) = \langle b| \text{ (given } c_\alpha, c_\beta \text{ and } \langle a|) \tag{13}$$

Eqn. (8) has also another reverse, denoted by the operator $\overleftarrow{Z}^t$, connecting c and $\langle b|$ to give $\langle a|$, the corresponding matrix being $|Z^t|$. Similar to (13), we can write this relation as

$$(c \mid \overleftarrow{Z^t} \mid b) = \langle a| \text{ (given } c_\alpha, c_\beta \text{ and } \langle b|) \tag{14}$$

Since reversals of matrix operators representing binary relations yield only vectors contained in the Boolean algebra, the algebra is closed with respect to these also. On the other hand, the algebra is not closed when the Boolean operations $\oplus$ and $\otimes$ are reversed. This is not discussed here, but the consequences are indicated for the particular case of BA-2, in paper[1], and its generalization is indicated there.

4. Vector-Matrix Formalism Applied to Propositional Calculus

This aspect has been discussed in the previous paper[1] and will not be considered in detail here. We shall only comment on the fact that BA-1 readily represents the classical sentential calculus as discussed in standard books on logic[3,4], while BA-2 is needed for its extended form, namely SNS. It may be mentioned that the matrices discussed in the previous section for the theory of relations are in general rectangular ($m \times n$), but for Boolean algebra of genus-n (BA-n), the number of states possible is n for all entities, and hence it is representable by n-element vectors and $n \times n$ square matrices. The other formulae in the previous two sections are unaltered.

SNS has four states, T, F, D, X corresponding to the four different vectors (1 0), (0 1), (1 1) and (0 0) of BA-2. This set of states is closed with respect to 2×2 Boolean matrix multiplications, *i.e.* logical relations of the forward type. From equation (9a–d), it is easy to show that reverse relations corresponding to all the 16 matrix operators also lead to one of the four states, represented by these vectors. This can be taken as a *proof of the closure of propositional logic.*

On the other hand, when the standard Boolean algebraic connectives $\oplus$ and $\otimes$ are applied to SNS, they lead again only to one of the four states, as shown in Table 3. In this case, when reverse operations are considered for these two logical operators which have been given the name "unanimity" (**U**) and "vidya" (**V**), something not contained in BA-2 is produced. These operators are the same as **O** and **A** for PC with BA-1 representation, but are quite different when applied to the *four* states of SNS. In fact, U and V are not the same as "or" and "and" in SNS, and lead to 4×4 truth tables and produce states outside even BA-2 on reversal.

In order to illustrate how such a state outside BA-2 occurs when a relation a Vb = c is reversed, we shall consider the case where c = T and a = T, and we ask, "What is c $\overleftarrow{V}$a = b?" This is readily answered by looking at Table 3, and we get the state "T or D, but not F". Similarly, we can also obtain "F or D but not T", from the application of the reverse $\overleftarrow{V}$ operator. If these two are combined by the connective V, we obtain the state "D, but neither T nor F", whose complement is "T or F, but not D". We thus obtain four new states in addition to the four standard states T, F, D, X isomorphic to BA-2. These give a total of eight ($= 2^3$) states, which can be shown to be isomorphic to the 8 states of BA-3.

Therefore, if the Boolean operators $\oplus$ and $\otimes$ are reversed in BA-2, we get not only states occurring in BA-2, but also four others, leading to a complete set of states of BA-3.

TABLE 3.

Truth tables for unanimity (U) and vidya (V) worked out using the algebra of BA-2

(a) $a'Va'' = a^*$

a' \ a''	T	F	D	X
T	T	X	T	X
F	X	F	F	X
D	T	F	D	X
X	X	X	X	X

$^* V \equiv \otimes$

(b) $a'Ua'' = a^\dagger$

a'' \ a	T	F	D	X
T	T	D	D	T
F	D	F	D	F
D	D	D	D	D
X	T	F	D	X

$^\dagger U \equiv \oplus$

(*a*) *Property of closure of Boolean algebras including reverse operations*:

Just as BA-2 with $3(=2^2-1)$ non-impossible elements leads, by binary reversal of the operation of Boolean $\otimes$, to BA-3 with 2^2-1 $(=3)$ generators and $7(=2^3-1)$ non-impossible elements, the process can be extended to give a series of BA-n's successively, with $n=7$, $n'(=2^7-1)$, $n''(=2^{n'}-1)$, etc., generators, producing an infinite sequence of Boolean algebras (BA-n) with $n = 2, 3, 7, n', n''$, etc., going upto an infinite value for n. Hence, if "reversal" of all relations is an admissible operation (similar to "inverse" in ordinary algebra, group theory, etc.), then, Boolean rings (BA-n) with $n = 2, 3, 7\ldots$ are interrelated, and any one leads necessarily to the next higher one, on including the results of reversing the relations in the earlier one.

Thus, if a completely closed Boolean algebraic system is at all possible, it must necessarily have an infinite number of elements. This is similar, for example, to the set of positive integers which has no last member, although every member of it can be described and mathematically utilized, in principle. We believe that this new concept of *non-closure of Boolean algebras of finite order* (2^n), has not been pointed out earlier in connection with studies on Boolean algebra.

However, if only matrix reversals are demanded (but not reversals of the Boolean operations $\oplus$ and $\otimes$ in BA-n), then the *Boolean algebra generated by n basic vectors in our vector-matrix formalism is complete and closed.* This general theorem for BA-n has its repercussions in showing that BA-2 and BA-3, which can represent, according to us, PC and QPL of logic, are complete and closed, even including unary reversal and binary matrix reversal. This theorem *regarding BA-n indicates that logic as it is normally* understood is essentially a closed system. The only way it is open and leads on to a higher order from any BA-n is *via* the reversal of the forward operations of $\oplus$ and $\otimes$ in that BA-n system.

Thus Boolean algebra, mathematical logic, and Boolean vector-matrix formalism, are all isomorphic with one another and many theorems regarding logic can be derived from the representation—and this makes thinking and logical analysis much more easy.

5. ESSENTIALS OF QPL WITH REFERENCE TO BA-3

(*a*) *Standard QPL states*:

The two quantifiers that are used in the standard theory of QPL are "for all" $(\forall x)$ and "there exists" $(\exists x)$. These "quantify" the sentential predicate, which we may denote, for the variable x, by $(\mathrm{s}x)$ with s standing, in general, for "sentence". There is also a third entity, that is applied in front of the quantifier, which has two states, corresponding to an affirmation or a negation of the quantifier. We suggest the name "sign" for this and use the symbol $\urcorner$ to denote. negation—*e.g.* $\urcorner(\forall x)$ $(\mathrm{a}x \Longrightarrow \mathrm{b}x)$ or $\urcorner(\exists x)$ $(\mathrm{a}x \& \mathrm{b}x)$. The negation of the sentence s, where necessary, is indicated by the PC or SNS negation symbol $\neg$ before s. However, the eight quantified forms that are obtained using $(\forall x)$, $(\exists x)$ $\neg$ and $\urcorner$ are not independent. They form four pairs of equivalent statements, as given in (15a to d).

$$(\forall x)(\mathrm{s}x) = \urcorner(\exists x)(\neg \mathrm{s}x) \qquad (15a)$$

$$\urcorner(\forall x)(\mathrm{s}x) = (\exists x)(\neg \mathrm{s}x) \qquad (15b)$$

$$(\forall x)(\neg \mathrm{s}x) = \urcorner(\exists x)(\mathrm{s}x) \qquad (15c)$$

$$\urcorner(\forall x)(\neg \mathrm{s}x) = (\exists x)(\mathrm{s}x) \qquad (15d)$$

The quantifier "for none" (Φx), standing for "not there exists" *viz.* $\urcorner(\exists x)$, is used quite often, and this suggests that we must coin a symbol also for "not for all", and we employ (Λx) for this, standing for $\urcorner(\forall x)$.

We have used two different symbols $\urcorner$ and $\neg$ in the above examples, and these two have entirely different algebraic properties, although they both have only two states—namely "yes" and "no". The former negates only the state of the quantifier—*e.g.* $\urcorner(\forall x)$ means "not for all" and not "for all, not". The latter acts only on sx. The two are interrelated (see later for precise details). Thus we have

$$\neg(\forall x)(sx) = (\exists x)(\neg sx) \tag{16a}$$

$$(\forall x)(\neg sx) = (\Phi x)(sx) \tag{16b}$$

The second of these demands that *every* x has the property "not s x", while the first only requires that there are *some* x's having the property "not sx".

A little reflection will show that the state $(\forall x)$ is implicitly assumed in every statement of propositional calculus, without mentioning any particular quantifier state with reference to it. Thus "men are mortal" is equivalent to saying, "*all* men are mortal". We will find this property of $(\forall x)$ that it is an unstated quantifier of propositional calculus very useful in deriving relations between statements in PC and statements in QPL.

(b) Boolean vector representation of standard QPL states:

The statements of QPL in the standard form, such as those in (15) and (16) can be represented *via* BA-1, BA-2 and BA-3 vectors as follows:

(i) We use SNS for sx which is a statement in PC, and this requires two symbols s_α and $s_{\bar{\beta}}$, written simply as $(\alpha\ \beta)$, for defining it, with four truth values T, F, D, X. The negation operator $\neg$ for sx is the SNS operator N corresponding to the BA-2 matrix $|N|$ (see ref. 1).

(ii) As will be clear from the discussions in the next section, we find an absolute need for BA-3 to represent the quantifier. Although we have needed so far, in our discussions, only four quantifier states $\forall$, $\exists$, Φ, Λ, they do not form a complete Boolean algebra of any genus, and are properly represented only in BA-3. Preliminary studies indicated that their properties are representable by the BA-3 vectors:

$$\forall = (1\ 0\ 0),\ \exists = (1\ 1\ 0),\ \Phi = (0\ 0\ 1),$$
$$\Lambda = (0\ 1\ 1) \tag{17}$$

Simultaneously, the other negation operator $\neg$ is the BA-3 complementation operator ($\sim$), whose effects we denote by a superscript c, or as will be seen in the next section, by the operator M. The effects of this operator is to change 1 to 0 and 0 to 1 in all the three vector components q_γ, q_δ, q_ϵ, of the quantifier q, written as $(\gamma', \delta', \epsilon')$. It is readily verified that

$$\forall^c = \Lambda,\ \Lambda^c = \forall,\ \exists^c = \Phi,\ \Phi^c = \exists \tag{18}$$

(The reason for the primes in $\gamma', \delta', \epsilon'$ is to distinguish the quantifier in the standard form from that in the canonical form which will be defined in subsection (c) below.)

(iii) We shall use an one-element vector (ζ) to denote the "sign" of the standard form, associated with the negation symbol $\neg$. Thus $\zeta = 1$ indicates affirmation and $\zeta = 0$ indicates negation of the quantifier.

With the above definitions, a quantified state q' in QPL, *expressed in the standard form adopted in all textbooks*, requires six parameters (q_ζ), $(q'_\gamma\, q'_\delta\, q'_\epsilon)$ $(q_\alpha\ q_{\bar{\beta}})$ to represent it. In this, every one of the components from q_α to q_ζ is a Boolean variable in BA-1, having only two possible values 1 and 0. Where convenient, we shall represent the state in the "standard" form by $(\zeta)(\gamma'\,\zeta'\,\epsilon')(\alpha\ \beta)$, in which the primes for γ, δ, ϵ are used to distinguish this from the "canonical" form $(\gamma\ \delta\ \epsilon)$ discussed in the next subsection.

It would appear, from the 6-element description of q in the standard format, that there are $2^6 = 64$ possible different quantified states. Actually this is not so and only eight of them are distinct, out of which four cover all the standard quantified states employed in the literature for standard QPL. These features are described in the next subsection 5(c), where we shall use the term "extended" QPL (EPL) if it is necessary to draw attention to the extra four non-standard states of BA-3 specifically.

(c) Interrelations between standard QPL states:

Out of the 64 possible standard forms, 16 are especially interesting in that the treatment of quantified predicate logic in the literature is based only on these just as propositional calculus uses only two states T and F in the standard literature, while we find the need for two more states D and X, when reverse relations are taken into account. These 16 can be obtained by taking the four quantifier states $(\forall x)$, (Λx), $(\exists x)$ (Φx) and attaching, in front and after the quantifier symbol, the BA-1 symbols $\neg$ and $\neg$ where necessary. Table 4 below lists these 16 different forms in QPL, of which there are four equivalent forms of each of these states, which are interrelated to one another by four equivalence operators, called "modifiers" $\mathcal{E}_E$, $\mathcal{E}_N$, $\mathcal{E}_M$, $\mathcal{E}_L$.

In each row of this table, there is just one entry which does not have negation either for the sign or for the sentence. In this particular form (enclosed in a box), the affirmative sense occurs for the quantifier and the sentence, and hence we abstract the quantifier part alone of this, and use it for the name of the full quantifier predicate state, which is represented, in the standard form, by all the four entries in that row, and which are logically equivalent to one another. This is shown in the second column of the table which requires only three components γ, δ, ϵ. We name this

TABLE 4

The sixteen different forms in standard QPL obtainable from $\forall$ *and* $\exists$ *and their interrelationships*

Operator	Canonical form	Modifier used for translation and standard form			
		ε_E	ε_N	ε_M	ε_L
	$(\gamma\ \delta\ \epsilon)$	$(\zeta)\ (\gamma'\ \delta'\ \epsilon')\ (\alpha\ \beta)$	$(\zeta)\ (\epsilon'\ \delta'\ \gamma')\ (\beta\ \alpha)$	$(\zeta^c)\ (\gamma'^c\ \delta'^c\ \epsilon'^c)\ (\alpha\ \beta)$	$(\zeta^c)\ (\epsilon'^c\ \delta'^c\ \gamma'^c)\ (\beta\ \alpha)$
E	$(\forall x)$ (1 0 0)	$\boxed{(\forall x)\ (\mathrm{s}x)}$ (1) (1 0 0) (1 0)	$(\Phi x)\ (\neg \mathrm{s}x)$ (1) (0 0 1) (0 1)	$\neg(\Lambda x)\ (\mathrm{s}x)$ (0) (0 1 1) (1 0)	$\neg(\exists x)\ (\neg \mathrm{s}x)$ (0) (1 1 0) (0 1)
N	(Φx) (0 0 1)	$(\forall x)\ (\neg \mathrm{s}x)$ (1) (1 0 0) (0 1)	$\boxed{(\Phi x)\ (\mathrm{s}x)}$ (1) (0 0 1) (1 0)	$\neg(\Lambda x)\ (\neg \mathrm{s}x)$ (0) (0 1 1) (0 1)	$\neg(\exists x)\ (\mathrm{s}x)$ (0) (1 1 0) (1 0)
M	(Λx) (0 1 1)	$\neg(\forall x)\ (\mathrm{s}x)$ (0) (1 0 0) (1 0)	$\neg(\Phi x)\ (\neg \mathrm{s}x)$ (0) (0 0 1) (0 1)	$\boxed{(\Lambda x)\ (\mathrm{s}x)}$ (1) (0 1 1) (1 0)	$(\exists x)\ (\neg \mathrm{s}x)$ (1) (1 1 0) (0 1)
L	$(\exists x)$ (1 1 0)	$\neg(\forall x)\ (\neg \mathrm{s}x)$ (0) (1 0 0) (0 1)	$\neg(\Phi x)\ (\mathrm{s}x)$ (0) (0 0 1) (1 0)	$(\Lambda x)\ (\neg \mathrm{s}x)$ (1) (0 1 1) (0 1)	$\boxed{(\exists x)\ (\mathrm{s}x)}$ (1) (1 1 0) (1 0)

symbol as the "canonical" form of the state, representable by just one three-element Boolean vector $(\gamma\ \delta\ \epsilon)$ of BA-3. the primes on $(\gamma'\ \delta'\ \epsilon')$ which occurs in the standard form have been removed while expressing the QPL term in the canonical form.

The four equivalence operators $\mathcal{E}_E$, $\mathcal{E}_L$, $\mathcal{E}_M$, $\mathcal{E}_N$, interconvert the standard forms that are equivalent into one another. They form a group isomorphic to the well-known "Four-group", with the following properties:

$$\mathcal{E}_E^2 = \mathcal{E}_L^2 = \mathcal{E}_M^2 = \mathcal{E}_N^2 = \mathcal{E}_E; \tag{19a}$$

$$\begin{aligned} \mathcal{E}_L \mathcal{E}_M &= \mathcal{E}_M \mathcal{E}_L = \mathcal{E}_N; \\ \mathcal{E}_M \mathcal{E}_N &= \mathcal{E}_N \mathcal{E}_M = \mathcal{E}_L; \\ \mathcal{E}_N \mathcal{E}_L &= \mathcal{E}_L \mathcal{E}_N = \mathcal{E}_M \end{aligned} \tag{19b}$$

In fact the four canonical states "for all" ($\forall$), "for none" (Φ), "not for all" (Λ) and "there exists" ($\exists$), which are themselves related by QPL operators, E, N, M, L, are shown in Column 1 of Table 3. If now the above four states represented by canonical vectors in Column 2 of Table 4 are added and multiplied by Boolean operators $\oplus$ and $\otimes$ we get four more states. The names of these, as well as the new symbols coined for them, are shown in figure 1.

They are listed in Table 5 as the Boolean vectors comprising the representation of the 8 states of EPL (EPL standing for "Extended" Quantified Predicate Logic). These have been given the symbols q(1) to q(8) in that table, and the states into which each one goes, when operated by the EPL operators, E, N, M, L, are also shown therein. In fact Table 5 is a complete truth table for these operators.

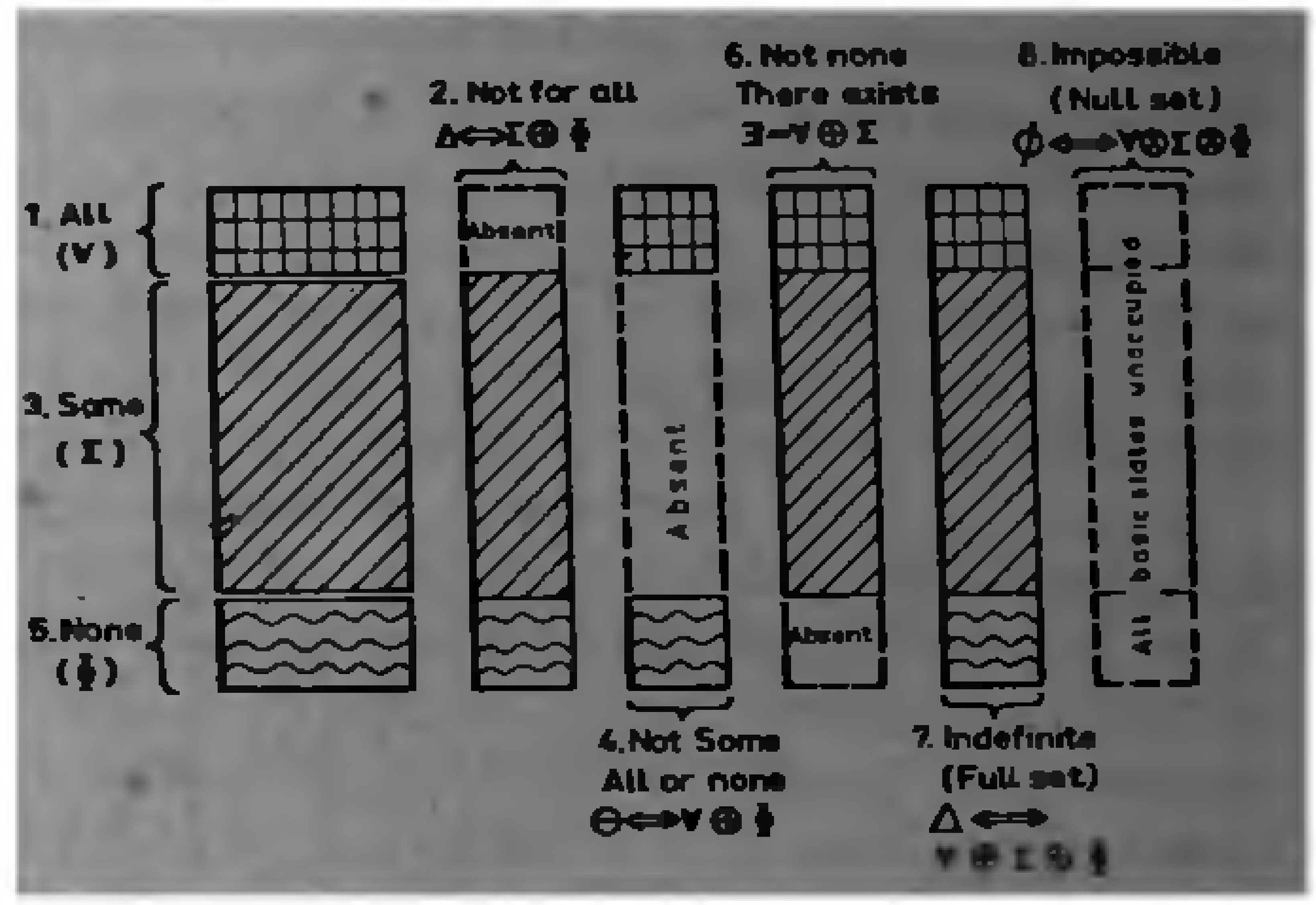

Figure 1. Schematic representation of the 8 possible states in EQPL.

The four new states of EPL that have been thus obtained have the following properties. The most interesting is the new *basic* state Σ (for some), in addition to the two standard ones V (for all) and Φ (for none). The state Σ $q(8)$ means that *only* some exist, but not all or none. If we take the complement of "some", we get "all or none" (q(4)) indicated by the symbol θ. Finally, we have the indefinite state (q(7) or Δ) which is obtained by adding all the three basic states $\forall$, Σ and Φ, and a statement in this indefinite

TABLE 5

Boolean vectors for the 8 states of QPL†

Sl. No. of state	Name of quantifier	Boolean vector $(\gamma\ \delta\ \epsilon)$	Symbol of state when operated by			
			E	N	M	L
q(1)	For all	(1 0 0)	$\forall$	Φ	Λ	$\exists$
q(2)	Not for all	(0 1 1)	Λ	$\exists$	$\forall$	Φ
q(3)	For some	(0 1 0)	Σ	Σ	Θ	Θ
q(4)	All or none	(1 0 1)	Θ	Θ	Σ	Σ
q(5)	For none	(0 0 1)	Φ	$\forall$	$\exists$	Λ
q(6)	There exists	(1 1 0)	$\exists$	Λ	Φ	$\forall$
q(7)	Indefinite	(1 1 1)	Δ	Δ	$\varnothing$	$\varnothing$
q(8)	Impossible	(0 0 0)	$\varnothing$	$\varnothing$	Δ	Δ

Each of the four pairs consists of a set of two mutually complementary states—*e.g.* $\forall^c = \Lambda$, $\Sigma^c = \theta$, $\Phi^c = \exists$ $\Delta^c = \phi$

state (Δx) (s x) has absolutely no logical information content for the term denoted by sx. However, anticipating the discussion in the next section, if the vidya operator ($\otimes$) is applied between this vector (1 1 1) and one of the other seven states $(\gamma\,\delta\,\epsilon)$, then the other state $(\gamma\,\delta\,\epsilon)$ will be the output. Similarly, the "impossible' state $q(8)$ ($\varnothing$) is obtained by taking the intersection of all the three basic states $\forall$, Σ and Φ. Since they are mutually non-overlapping, their intersection will be the null set $\varnothing$, corresponding to the "impossible" state in logic and denoted by the Boolean vector (0 0 0) of EPL.

Before considering the effects of 3×3 matrix operators and the Boolean symbolic operators $\oplus$ and $\otimes$ on the eight BA-3 vectors representing (in the canonical form) the 8 states of EPL, we shall classify the 64 possible quantified terms in the standard notation, including the 16 of Table 5, which can occur in EPL. Thus,

(i) Quantifier is one of $\forall$, Δ, Λ, $\exists$ of QPL and s = T or F, sign = 0 or 1. These 16 were considered in Table 3, and lead only to 4 states.

(ii) Quantifier is one of the four new states of EPL, and s = T or F, sign = 0 or 1. These also lead only to four standard states (Σx) (s x), (Θx) (s x), (Δx)(s x) and $(\varnothing x)$ (s x), each of which produces a set of two equivalent standard forms, by the application of the modifers $\mathcal{E}_E$, $\mathcal{E}_N$, $\mathcal{E}_M$, $\mathcal{E}_L$.

(iii) Quantifier is any one of eight, sign either "yes" or "no" (0 or 1) but s = D. All sixteen of these lead to the same canonical state $\Delta(\equiv (\Delta x)$ (s x)).

(iv) Quantifier is any one of eight, sign either "yes" or "no" (0 or 1), but s = X. Again all sixteen lead to the same canonical state $\varnothing(\equiv \varnothing x)$ (s x)).

Just as with the set of states (i), the properties of those in (ii), (iii), (iv) above have been formulated by us from an examination of the logical contents, and equivalences, of the relevant standard forms. The effect of s = D can be explained by saying that if the statement s has a doubtful state and can give no information, then it converts itself automatically into the universally doubtful quantified state Δ. Similarly s = X leads straightaway to the impossible quantified state $\varnothing$. Further details are reserved for a more extensive presentation elsewhere.

(d) *Canonical states of EPL and their use with connectives:*

Since the eight canonical states, isomorphic to the 8 states of BA-3, cover all the standard forms of terms in EPL, we shall only consider connective operators (unary and binary) which interconnect these *canonical* states. While doing this, we shall indicate how all the well-known connectives of quantified predicate logic are covered and their properties (as envisaged in standard presentations on the subject) are all incorporated in our formulation. We shall first indicate how a standard form a′ is converted to its canonical form a and how a unary connective Z is applied to it to obtain the canonical form b of the resulting term, which can then be modified into the required standard form.

This is illustrated by an example in Table 6 (a, b, c). The problem to be solved can be stated in words as follows:

We are given that "For all x, a x is true implies that there exists a y such that by is true" (Step (b)). For this relation, the input is "There does not exist any a x that is false" (Step (a)), and we are asked to find out, given the quantifier state ($\forall y$) of the input, what is the nature of b y, and the sign of the quantifier (Step (c)).

The three steps involved are indicated schematically in Table 6, the most important of which is the representation of the connective involved, namely "implies" in Step b. Anticipating the form of this (namely one of the sixty four 3×3 matrices for "implies"—see Section 6), we state that it is I (1, 6) in the present case, the indices 1 and 6 standing for $\forall = q(1)$ and $\exists = q(6)$, which are connected by it. Since this BA-3 matrix requires that both the input and output vectors are in BA-3, *i.e.* the canonical form, Step (a) applies the "canonizer" first to convert the input from the standard form to the canonical form. The algorithm for the canonizer is given in Part A of Table 6 in Section 6(b). Similarly, when the output comes out in the canonical form, the "standardizer" of Step (c) converts it into the required standard form. The algorithm for the standardizer is given later in Table 6, Part B.

Although this example does not involve all the intricacies related to problems of this kind, it gives the essence of our procedure, and we obtain the required result in Step (c)—namely "Not for all y is the statement by false".

The operator I(1, 6) is a 3×3 Boolean matrix. Similarly, other connectives such as "and", "or", "nand" etc., of EPL are also matrix operators. These are discussed in Section 6(c), where their nature is derived physically by inspecting their expected logical nature. In doing this, we apply the procedure normally adopted in theoretical physics—namely of examining the properties of the connective as it is considered in standard QPL, and then giving them a mathematical interpretation using BA-3 for EPL (which includes QPL). In this sense, our method of approach is similar to the use of algebraic formulae and equations in Cartesian analytical geometry, for solving problems in pure geometry. On the other hand, standard treatments of logic via theorems follow

TABLE 6

Example of a QPL sentence in an argument implemented via canonical terms and connectives

Description of sub-step	Logical content and Boolean algebraic representation
(a) Input canonized	$\neg(\exists x)(\neg a x) \xrightarrow{C} (\forall x)(a x)$ $(0)(1\ 1\ 0)(0\ 1) \xrightarrow{C} (1\ 0\ 0)$ (Standard form) (Canonizer) (Canonical form)
(b) Canonical connective applied	$\forall(x) \Longrightarrow \exists(y)$ $(1\ 0\ 0)\ \ \mathrm{I}(1, 6) = (1\ 1\ 0)$; $\mathrm{I}(1, 6) = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ (Canonical) (Connective) (Canonical)
(c) Output standardized	$\exists(y)(b y) \xrightarrow{S} \neg(\forall y)(\neg b y)$ $(1\ 1\ 0) \xrightarrow{S} (0)(1\ 0\ 0)(0\ 1)$ (Canonical form) (Standardized) (Standard form)

the method of Euclidean geometry in deriving proofs with existence conditions rather than giving a technique for working out problems. We belive that our method will solve problems very readily as will be shown in Section 7.

References are all given at the end of Part II.

ANNOUNCEMENT

NATIONAL SYMPOSIUM ON MICROBIAL ENERGY : PRODUCTION & CONSERVATION

The Department of Microbiology, G. B. Pant University of Agriculture and Technology, Pantnagar is organizing a National Symposium on "MICROBIAL ENERGY: PRODUCTION & CONSERVATION" on May 26–28, 1983, under the joint sponsorship of the Department of Science and Technology, Government of India and G. B. Pant University of Agriculture and Technology. The major areas of discussion in this symposium are: (1) Fermentation Technology, (2) Recycling of Wastes and Methanogenesis, (3) Photoproduction of hydrogen and biophotolysis, (4) Ecological and Social Implications, and (5) Engineering and Economics Implications. Some travel support to deserving Young Scientists under 32 years of age is available.

For further details write to:
Prof. Bhavidish N Johri
Convenor
Department of Microbiology
G. B. Pant University of Agriculture & Technology
Pantnagar (Nainital) UP 263 145.