MEMORANDUM
RM-5428-PR
DECEMBER 1967

ANSWERING QUESTIONS BY COMPUTER: A LOGICAL STUDY

J. L. Kuhns

PREPARED FOR:
UNITED STATES AIR FORCE PROJECT RAND

The RAND Corporation
SANTA MONICA • CALIFORNIA

MEMORANDUM
RM-5428-PR
DECEMBER 1967

ANSWERING QUESTIONS BY COMPUTER: A LOGICAL STUDY

J. L. Kuhns

This research is supported by the United States Air Force under Project RAND — Contract No. F44620-67-C-0045 — monitored by the Directorate of Operational Requirements and Development Plans, Deputy Chief of Staff, Research and Development, Hq USAF. RAND Memoranda are subject to critical review procedures at the research department and corporate levels. Views and conclusions expressed herein are nevertheless the primary responsibility of the author, and should not be interpreted as representing the official opinion or policy of the United States Air Force or of The RAND Corporation.

The RAND Corporation

THE RAND CORPORATION 1700 MAIN STREET, SANTA MONICA, CALIFORNIA 90406

ERRATA

RM-5428-PR, Answering Questions by Computer: A Logical Study, by J. L. Kuhns, December 1967.

The Summary of RM-5428-PR has several statements which can be seriously misconstrued if the reader is oriented to the modern theory of algorithms. To avoid this the following changes should be made:

On page v, in lines 23-28, read:

> . . . How can a computer recognize "reasonable" symbolic questions? It turns out that a logical characterization of "reasonable" symbolic questions is possible and that this characterization can be expressed in terms of machine-recognizable sufficient criteria.

On page vi, in line 6, correct the error "unreasonable" by replacing it with "reasonable".

On the same page, in line 10, replace "solve" by "study". (The exact sense in which the identification problem is solved is given by Theorems 23, 24, and the remarks following Theorem 21.)

Finally, for the reasons given above, the following additional changes must be made:

On page 9, in line 14, replace "solve" by "study".

On page 11, in line 12, read: "The first problem is the development of an . . ."

PREFACE

This work was developed in the past year as part of The RAND Corporation's Cybernetic Data Research Project--a project for the implementation of the computerized information system proposed by R. E. Levien and M. E. Maron in their RAND Memorandum, "Relational Data File: A Tool for Mechanized Inference Execution and Data Retrieval" [1].

The present study is one of a series of reports to come from this project.† It is specifically motivated by certain technical problems dealing with the extraction of data from the file. The results, however, are applicable to the general problem of question-answering by computer.

To my colleagues at RAND I wish to express my thanks for stimulation and encouragement. In particular, my thanks go to N. D. Cohen, R. E. Levien (project leader), G. Levitt, M. E. Maron (formerly of RAND and former project co-leader), and R. Mattison. I am especially indebted to J. Economos for the many hours of clarifying discussion, criticism, and review of portions of the manuscript.

---

†For further information see M E. Maron, "Relational Data File I: Design Philosophy," Proceedings of the Third Annual National Colloquium on Information Retrieval, May 12-13, 1966 (also P-3408, The RAND Corporation, July 1966); R. E. Levien, "Relational Data File II: Implementation," Proceedings of the Third Annual National Colloquium on Information Retrieval, May 12-13, 1966 (also P-3411, The RAND Corporation, July 1966); R. E. Levien and M. E. Maron, A Computer System for Inference Execution and Data Retrieval, The RAND Corporation, RM-5085-PR, September 1966.

BLANK PAGE

SUMMARY

The problem of computerized question-answering is seen to involve a two-step procedure: first, transforming the question into a certain sentential formula of the predicate calculus (called the symbolic question); second, generating the answer by calculating (what we shall call) the value set of the transformed question.

The main purpose of the present work is to investigate the second, or answering, process. The process is to output the answer by acting on the symbolic question and the data base for the question-answering system. The data base consists of a dictionary of descriptive names together with a file of elementary sentences. The value set of a symbolic question containing free variables (e.g., those which stem from natural-language questions such as 'What books has Scott written?') is the list of names which when substituted for the free variables yield a "true" sentence on the data base. For a symbolic question without free variables (e.g., one which stems from natural-language questions such as 'Did Scott write Waverley?') the value set, or simply the value, is an expression indicating the truth value.

A key problem in calculating value sets is the identification of questions that are "unreasonable"--for example, the question 'Who did not write Waverley?'. How can a computer determine that such a question is unreasonable? It turns out that a logical characterization of "unreasonable" symbolic questions is possible and that this characterization can be expressed in terms of machine-recognizable criteria.

The characterization is developed by introducing the concept of definite formula. A formula is definite if, for any data base, the value set of the formula is unchanged when a new descriptive name is added to the dictionary of the data base. The rather vague notion of "unreasonable" question is now replaced by the specific notion of definite formula. Among the definite formulas is a class of particular interest--the proper formulas. These formulas, because of their structure, are especially suitable for machine processing. We solve the problem of identifying the proper formulas and calculating their value sets.

Finally, the important class of definite but improper formulas is studied. The approach is to transform these into proper equivalents. Those formulas for which this transformation can be done are called admissible. It is conjectured that every definite formula is admissible. The admissibility of definite formulas without quantifiers is proved. For definite formulas with quantifiers, limited, but useful, results are obtained.

CONTENTS

1. INTRODUCTION

1.1 PRELIMINARY REMARKS

This Memorandum is part of a general study of the problem of answering questions by computer. The problem is seen to involve a two-step procedure: first, transforming the question into a certain expression of the predicate calculus (called the symbolic question); second, generating the answer by calculating (what we shall call) the value set of the transformed question.

In a subsequent report we will deal with the first step of this procedure. The present study is addressed to the answering process. The study was motivated by the data retrieval system proposed by Levien and Maron [1].

Sections 1 and 2 give an informal discussion of the problem and present some background material. The formal development of the theory is given in Secs. 3-6.

1.2 REPRESENTATION OF DATA

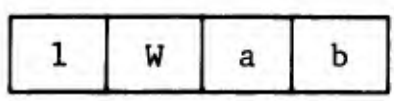
The data base for the information system consists of bibliographic and related information; i.e., facts about who wrote what paper, with what organization someone is affiliated, who sponsors what work, etc. Levien and Maron [1] point out that such data can be represented most conveniently in terms of the theory of relations. For example, consider the simple declarative sentence 'R. Carnap wrote Meaning and Necessity'. This concerns a relation between a person and a written work. In the notation of mathematical logic this relation can be denoted by 'W' followed by two argument positions. Letting

'a' and 'b' be symbolic translations of 'R. Carnap' and 'Meaning and Necessity', respectively, the sentence then receives the symbolic translation

Wab (1)

'W' is called a two-place predicate and designates a relation; 'a' and 'b' are called individual constants and designate individuals.†

The computer file for storing such data has a structure that matches this symbolic translation. First, the English expressions are translated into computer words (analogous to 'W', 'a', 'b'). Next, the three computer words are stored in the file (in this case, a disk file). Finally, a fourth computer word is used to identify the sentence so stored (analogous to the identification '(1)' above). Thus we have a data line of the design



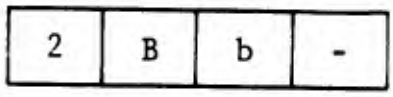
Consider next the sentence 'Meaning and Necessity is a book'. This concerns the written work Meaning and Necessity and a property, namely, that of being a book. This property is designated by a one-place predicate 'book'. Thus the sentence receives the symbolic translation

Bb (2)

---

†We mean "individuals" in a broad sense, including persons, things, books, organizations, etc.

where 'B' is a translation of the predicate 'book'. This leads to a data line in the file having the design



with no entry in the last place.†

The data base thus consists of two parts: 1) a dictionary of individual constants and predicates (called descriptive names); 2) a file of elementary sentences (certain strings of descriptive names).

Tables 1 and 2 give an example dictionary and an example file, respectively.

1.3 SYMBOLIC QUESTIONS AND VALUE SETS

We now turn our attention to the problem of interrogating the information system. Suppose the inquirer wishes to know

What books has R. Carnap written? (3)

The first step is to communicate this requirement to the system. There are several possibilities for the input

---

†The ordering of the words in the data line is not significant as long as the word types are machine recognizable. In fact, to facilitate our discussion we have described a slightly different arrangement from the one used in the actual file. The actual file has the relation name (two-place predicate) in the third position. This is comparable to the symbolic translation used in the theory of binary relations where (1) would be written as 'aWb'. This relational notation is extended to sentences such as (2) by writing them as a relation between an individual and a property.

Table 1

AN EXAMPLE DICTIONARY

a	R. Carnap
b	"Meaning and Necessity"
c	"The Aim of Inductive Logic"
d	"The Logical Syntax of Language"
e	H. Reichenbach
f	"Elements of Symbolic Logic"
g	B. Russell
h	"Principia Mathematica"
i	A. Whitehead
B	(the property of being a) book
P	(the property of being a) paper
W	wrote (the relation of authorship in the broad sense)

Table 2

AN EXAMPLE FILE

1	W	a	b
2	B	b	-
3	W	a	c
4	P	c	-
5	W	a	d
6	B	d	-
7	W	e	f
8	B	f	-
9	W	g	h
10	W	i	h
11	B	h	-

query. One possibility is to formulate it as a procedure of the kind described by Levien and Maron [1]. Another approach is to input directly the natural-language question. In either case (more obviously in the second), the input stage must be accompanied by a representation process that transforms the query into a form that can serve as a retrieval prescription. This form we call the symbolic question. The exact mechanism of this transformation will not concern us here. We are interested only in the result, and assume this to be a sentential formula in the predicate calculus.† For example, the sentential formula associated with (3) is

(Bx) AND (Wax) (4)

(The expression (4) can only be identified with (3) if we add a new symbol to indicate it is a question. A technique for symbolization proposed in the literature (Carnap [2], Reichenbach [3]) is to prefix the sign

(?x)

This then tells the receiver that what follows is to be processed in a certain way. Similarly, Levien and Maron [1] would begin their input query with 'EXTRACT (x)'. Thus (4) is not the question itself but the sentential formula which must be processed to answer the question.)

†The author has demonstrated the possibility of automatically transforming a rather general class of natural-language questions into symbolic questions. The demonstration rests on a computer program for doing this.

The formula (4) is made up of several logical "parts of speech": the descriptive names 'a', 'B', 'W'; the variable 'x'; the logical connective or operator 'AND';† and finally the parentheses--the logical signs of grouping.

Now (4) involves the variable 'x' in a certain intrinsic way. This is called a free variable. Formulas with such variables are called open sentential formulas. A closed sentential formula (or simply a sentence) is one that contains no free variables. If we substitute various descriptive names for 'x' in (4), we get a collection of closed sentential formulas each having the design '( ) AND ( )'. The sentences in the parentheses are, in this example, elementary (i.e., they do not contain operators). Some of these are in the file and some not. When both are found in the file, we call the entire formula true. For example, 'Bb' and 'Wab' are such a pair. The answer to the question is then given by those substitutions which make (4) a true sentence. We say this list of substitutions gives the value set of the formula.

Some questions lead directly to closed sentential formulas. For example:

Did Carnap write the book Meaning and Necessity? (5)

leads to the formula

(Bb) AND (Wab) (6)

---

†We shall develop our signs in terms of machine-readable characters; the traditional signs will not be used.

Thus we regard the question as a sentence to be affirmed. The answer must then give the truth value of (6). For this reason we extend the notion of value set to a closed sentential formula by defining its value set to be a symbol designating its truth value.†

So far everything seems simple enough. In the next section we show some complications.

---

†The notion of value set is completely analogous to that of extension [4]. However, there are technical distinctions which make it incorrect to use that term. The main distinction is that value sets are expressions or classes of expressions while extensions are not expressions but the designata of expressions. Moreover, it turns out that this distinction cannot be eliminated by shifting the discussion to the so-called object language [3] because we wish to use value sets as answers to questions. A detailed discussion of this problem will be given in a subsequent Memorandum.

2. THE PROBLEM OF CALCULATING VALUE SETS

2.1 A DIFFICULTY

Consider the question

Who did not write Meaning and Necessity? (1)

We would certainly regard this as unreasonable, but how is the computer to know this? Should the computer simply list all the names in the dictionary (except 'R. Carnap') or should it somehow prohibit the question?

The symbolic form of (1) is the formula

NOT(Wxb) (2)

The difficulty with this expression is that its value set depends on more than the value set of the component 'Wxb'; it depends on other names in the dictionary. That is, if we add a new name to the dictionary the value set of (2) will change. A formula without this objectionable property will be called definite.†

Other examples of indefinite formulas are easy to construct in terms of procedural requests. For example, the request

EXTRACT (x,y): x wrote Meaning and Necessity or
y wrote Meaning and Necessity (3)

†In the formal development of the theory (Sec. 4) we shall add another requirement so that the definitude of a formula will be independent of the truth or falsehood of any particular sentence.

leads to the formula

(Wxb) OR (Wyb) (4)

This is indefinite because any ordered pair of names gives a member of the value set providing only that 'a' is in the first or second position.

Let us now replace the rather vague notion of what constitutes a "reasonable" request by the specific notion of definite formula. The problem is then to invent algorithms for identifying definite formulas and calculating their value sets.

We shall approach this problem in the following way. First, we define a class of formulas which, by their structure, lend themselves to machine processing. These are called proper formulas, and we solve the problem for this class. The second step is to look at the definite but improper formulas. These we propose to transform to proper formulas.

2.2 PROPER FORMULAS

Before going further, let us look more closely at the structure of sentential formulas. First, recall that an elementary formula is one without an operator (i.e., without an expression such as 'NOT', 'AND', 'OR', etc.). Then, because of our rules for writing formulas, any formula that is not elementary has either a design similar to (2), where a singulary operator is followed by a formula enclosed in parentheses, or a design similar to (4), where two formulas enclosed in parentheses are separated by a

binary operator. The formulas enclosed in such sets of parentheses are called components.

Now the components of a formula are either elementary or have components themselves. Thus we can think of a sentential formula as having a tree structure. For example, the formula

NOT(NOT((By) AND (Wxy)))

can be represented as in Fig. 1. The top and interior vertices correspond to the logical operators, and the bottom vertices to elementary formulas. A component associated with a particular operator is represented by the largest subtree below the vertex labelled with the operator.

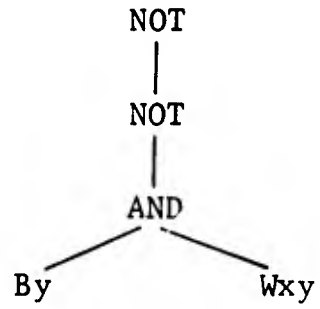


Fig. 1--Tree Representation of a Formula

From the point of view of machine processing the most desirable type of formula is one that is definite and whose value set stems only from the value sets of its components, which themselves are definite. But this transfers the problem to the components; therefore, we stipulate not only that these have the same property, but that their components do as well, and so on down the tree.

The class of formulas so singled out we call proper formulas. The exact definition is: a formula is proper if 1) it is elementary (and hence definite), or 2) it is definite and its components are proper.

The class of proper formulas is dependent upon the inventory of operators. It turns out that if we include an operator 'BN' (read as 'but not')† with the traditional operators for negation, conjunction, disjunction, implication, equivalence, and universal and existential quantification, then the proper formulas become interesting and useful.

2.3 ADMISSIBLE FORMULAS

The first problem we shall solve is to find an algorithm that 1) decides if any given formula is proper, and 2) if so, calculates its value set.

By the definition of propriety we see that, given such an algorithm, the machine can process a proper formula in a systematic way--starting with the elementary formulas and working up the tree.

We next propose to process a definite though improper formula by transforming it to a proper equivalent. This will now be explained.

Consider the formula

NOT(NOT(Wxb)) (5)

†This is analogous to relative complementation in the calculus of classes.

This is a simple example of a definite, but improper, formula. It is definite because it is co-valued with the formula

$$Wxb \tag{6}$$

It is improper because its component is indefinite.

A non-trivial example is given by the symbolic question stemming from

> Are all members of the editorial board of the Journal of the ACM mathematicians? (7)

Letting 'EB' denote the relation is on the editorial board of, 'M' denote the property (of being a) mathematician, and 'j' denote the Journal of the ACM, then by using the operators for universal quantification ('(Ax)') and implication ('IMP'), we represent (7) as

$$(Ax)((EBxj)\ IMP\ (Mx)) \tag{8}$$

The component of this formula is indefinite because any substitution for 'x' that makes the sentence 'EBxj' false makes the implication true. Therefore, the formula is improper. On the other hand, the formula has a truth value, and so it should be definite.

If a formula is co-valued† with (i.e., has the same value set as) a proper formula, we say it is admissible.

---

†This notion will be defined in the formal development (Sec. 5) to correspond to logical equivalence; i.e., if two formulas are co-valued they will be so independent of the contents of the file.

Thus (5) is admissible because it is co-valued with the elementary formula (6). It turns out that formula (8) is also admissible; it is co-valued with the proper formula

NOT((Ex)((EBxj) BN (Mx))) (9)

('(Ex)' is the existential quantifier; (9) can be read as 'it is not the case that somebody is on the editorial board of the Journal of the ACM but is not a mathematician'.)

We conjecture that any definite formula is admissible. We have proved this for definite formulas without quantifiers. For formulas with quantifiers, certain limited results have been obtained.

2.4 OUTLINE OF THE STUDY

This concludes the informal part of the study. In the remaining sections we give the formal exposition.

Section 3 gives the necessary logical preliminaries, developed in the following steps.

We begin with a group of expressions called elements. These are of two kinds: descriptive names (e.g., 'a', 'b', 'c', 'B', 'W'); and variables (e.g., 'x', 'y', 'z').

We next form finite sequences of elements, called basic sequences. For example, 'x,W', 'W,a,b', 'x,y', 'B,x' are basic sequences. (These correspond to the same expressions without commas, which are introduced only to represent machine-readable markings.) Certain of these are called elementary sentential formulas (e.g., 'x,W', 'W,a,b', 'B,x').

We then introduce some logical operators (e.g., 'AND', 'OR'), which are combined with the elementary formulas in certain ways to generate the class of sentential formulas.

Finally, we define the notion of the value set of a sentential formula. This notion is developed in terms of a stock of "true" elementary formulas specified as constituting the file (e.g., 'W,a,b' is so specified).

This now paves the way for the definition of definite formula and the particular class of them having desirable structural properties for machine processing, namely, the proper formulas. Section 4 presents the theory of these formulas and algorithms for their evaluation; it concludes with some remarks on the application of the theory to natural language and the logic of relations.

In Sec. 5 we study the class of admissible formulas (i.e., those formulas co-valued with proper formulas) and discuss certain results relating definite formulas to admissibility.† (See Fig. 2 below for an overall view of the major types of formulas.)

Section 6 presents guidelines and suggestions for implementing the theory, and concludes the study.

---

†Conjecture: Definite formulas are admissible.

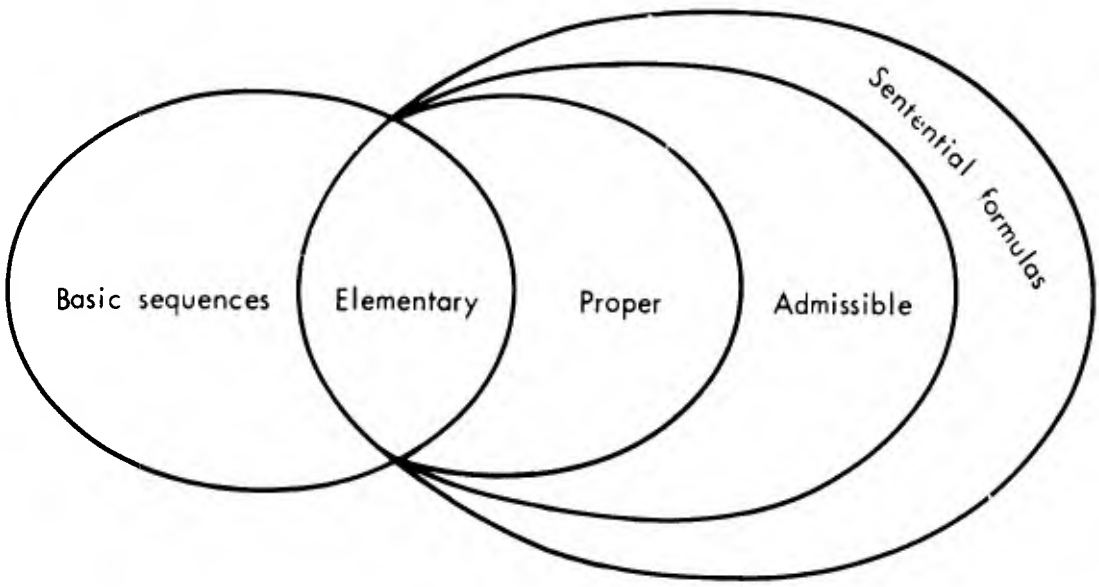


Fig. 2--Classification of Sentential Formulas

3. FORMULAS AND VALUE SETS

3.1 REMARKS ON NOTATION

In this section we discuss and classify certain machine-readable expressions. By 'machine-readable expression' we mean an expression built up from a character set consisting of the capital letters of the English alphabet, the digits, and the auxiliary signs ''', '*', Δ, ')', ',', '-', '$', '=', '(', '.', '+', '/'. In this list 'Δ' denotes the (machine-readable) blank. The phrase 'machine-readable' thus suggests the application of the theory to the computer representation of these expressions.

Suppose that a finite collection e of elements or words has been formed by certain serial listings of these characters. New expressions can then be formed from the elements and the characters themselves. To describe such expressions we use a notation based on the abbreviated notation used in Ref. 2. In this method we do not directly define an expression but instead specify its design or form with varying degrees of exactness. A class of expressions having a certain form will be given a name using entirely non-machine-readable characters (such as lower case letters, Greek letters, superscripts, underlining, etc.). The form of a new expression is then given by the serial listing of the forms of its parts. For example, if we say an expression has the form

e f

this means the expression consists of an element (expression of class e) followed by an expression of class f.

In addition to the class names, we also use machine-readable characters themselves to describe expressions. For example, if an expression is described as having the form

$$\underline{e}\ (\ \underline{f}$$

we mean that it consists of a sequence of three expressions: an element followed by an open parenthesis followed by an expression of class $\underline{f}$. Thus, '(' in this context is regarded as an abbreviation of 'open parenthesis'. Among all the machine-readable characters only two will not be used in this way: Δ, as explained above; the letter '0', which will be denoted by 'Ø'.

Whenever superscripts appear, they are always used as part of the name of a form; subscripts, however, are not so used.† Instead, subscripts are used as follows: A form given as

$$\underline{e}_1\ \underline{e}_1$$

means that an expression of that form consists of an element (expression of class $\underline{e}$) followed by its repetition. A form given as

$$\underline{e}_1\ \underline{e}_2$$

†Thus '$\underline{f}^2_1$' is to be read as the subscript '1' attached to the name '$\underline{f}^2$'.

means that an expression of that form consists of an expression of class e followed by another (possibly different) expression of class e.

Finally, to refer to the result of an operation on an expression, we use a functional notation. But this requires an argument name as well as a function name. In this case the form description is used for the argument name. For example, we may write

$$g(\underline{e}_1\ \underline{e}_2) = \underline{e}_1$$

to mean: if x is an expression of the form $\underline{e}_1\ \underline{e}_2$, then g(x) is the first element of x. The nature of the function will prevent any ambiguity.

3.2 SENTENTIAL FORMULAS

Elements. The first group of expressions we take as primitive, thus leaving open their mode of definition. We suppose only that they are machine-readable (as explained above) and that their definition does not conflict with the definitions to follow.

These primitive expressions--called elements, e--are divided into two classes: a set of descriptive names, d; and a set of variables, v. The class d is again divided into two classes: the individual constants, c; and predicates, p.

Example: Let

$\underline{d}$ = {'a', 'b', ..., 'i', 'B', 'P', 'W'}

$\underline{c}$ = {'a', 'b', ..., 'i'}

$\underline{p}$ = {'B', 'P', 'W'}

$\underline{v}$ = {'x', 'y', 'z', 'u', 'v'}

Basic Sequences. By a basic sequence, $\underline{b}$, we mean a finite sequence of one or more elements. The number of elements in a sequence is called its length; $\underline{b}^n$ is the set of all sequences of length n. More exactly:

D1.[†] The basic sequences, $\underline{b}$, of various lengths are developed as follows:

a. A $\underline{b}^1$ has the form

$$\underline{e}$$

b. If n > 1, then a $\underline{b}^n$ has the form

$$\underline{b}^{n-1}, \underline{e}$$

We have built in the comma as part of the definition; but this is not really essential. An alternate definition of $\underline{b}^n$ would be those functions with domain {1, ..., n} whose range is the set of elements. If we think of elements as

[†]Throughout, the more important definitions are marked 'D' and numbered consecutively. Similarly, theorems are marked 'T'.

certain computer words, a $\underline{b}^n$ becomes a "dimensioned subscripted variable" (in the FORTRAN sense).

Elementary Sentential Formulas. Among the basic sequences of length greater than one, we sort out some of special interest called elementary sentential formulas, $\underline{f}^n$. (Again, we use the superscript when we wish to specify the length.) We extend the definition to the case n = 1 by introducing a special expression 'T' (for 'tautology').

D2. The definition of $\underline{f}^n$ is:

a. An $\underline{f}^1$ has the form

T

b. If n > 1, then an $\underline{f}^n$ is a $\underline{b}^n$ in which at least one $\underline{d}$ occurs.

(Other requirements could be stated in order to give some rather weak formation rules of the predicate calculus. These would specify expressions that have a chance of being meaningful. Three requirements seem appropriate:

1) at least one $\underline{d}$ occurs in the $\underline{b}^n$;
2) a $\underline{c}$ does not occur in the first position;
3) the $\underline{e}$ that occurs in the first position does not occur elsewhere.

However, if we stipulate these requirements as part of the definition of $\underline{f}^n$, certain technical difficulties would arise later when we begin to substitute in sentential formulas (i.e., various nonsense could be generated by the violation of requirements 2) and 3)). A simple approach

would be, first, to modify requirement 2) to state that a p occurs in the first position; second, to allow only c's to be substituted. But such a solution would not permit us to calculate value sets containing predicates.† This is because the definition of such value sets requires the substitution of predicates for variables. On the other hand, if we wish to substitute predicates for variables and have sentential formulas be the result, then requirements 1), 2), and 3) must be accompanied by rather complicated rules of substitution.

Thus, we are led to adopt a weak definition of elementary sentential formula and simple substitution rules. We then put the burden of significance on a stock of elementary formulas (the file, explained below in Sec. 3.5) that satisfy the additional requirements 2) and 3).)

Examples: 'B,x,y', 'W,B,a', 'B,b', 'u,a,b' are elementary sentential formulas; 'x,y,z' is not.

Logical Operators. We widen our class of sentential formulas in the traditional manner by using two classes of logical operators: the singulary operators, α; and the binary operators, β. These are, of course, distinct from the class of elements. Exact specifications are given in the following definitions:

---

†In some instances value sets containing predicates stem from questions such as 'What relation holds between a and b?'. In a subsequent Memorandum we will discuss what could be considered "natural" answers to these questions.

D3a. An $\underline{\alpha}$ is an expression of one of the following forms:

NØT

(E$\underline{v}$)

(A$\underline{v}$)

(P$\underline{b}^n$)

where the $\underline{b}^n$ contains n distinct variables.

An (E$\underline{v}$) is an existential quantifier; an (A$\underline{v}$) is a universal quantifier. The operator (P$\underline{b}^n$) is a permutation operator. Its use will be explained later (Sec. 3.6).

D3b. A $\underline{\beta}$ is an expression of one of the following forms:

AND

ØR

BN

IMP

IFF

The $\underline{\beta}$'s correspond to conjunction, disjunction, the negation of implication (read 'BN' as 'but not'), implication, equivalence (read 'IFF' as 'if and only if'), respectively.

Sentential Formulas. We now define the sentential formulas, s, in terms of elementary sentential formulas and the logical operators.

Roughly speaking, we stipulate that the sentential formulas constitute the "smallest" class of expressions that can be generated according to the following rules:

1) An $\underline{f}^n$ is an $\underline{s}$;

2) An expression of the form

$$\underline{\alpha}(\underline{s}_1)$$

is an $\underline{s}$;

3) An expression of the form

$$(\underline{s}_1)\underline{\beta}(\underline{s}_2)$$

is an $\underline{s}$.

(In the formulas generated by 2) and 3) the sentential formulas enclosed in parentheses are called components of the sentential formula.†)

More exactly, let us call any class of expressions that contains the elementary formulas and is closed under the formation rules given by 2) and 3) above an E-class. For example, the collection of all expressions consisting of a finite number of machine-readable characters is such a class. We then define:

---

†Note that the scope of the operators is always specified. The well-known binding conventions to save parentheses do not concern us since these are for the convenience of humans and we are interested in machine processing. Even if formulas formulated with binding conventions were permissible input, still the machine must compile the formula, i.e., in effect, identify the components. This is what the parentheses do.

D4. The class of sentential formulas, $\underline{s}$, is the class of expressions common to every E-class.

A paraphrase of this definition gives the following inductive principle, which we shall later find useful.

T1. Let ρ be a certain property of expressions. If

1) every elementary formula has the property ρ,

and

2) for all expressions x and y with ρ, $\underline{\alpha}(x)$ and $(x)\underline{\beta}(y)$ have ρ,

then:

every sentential formula has the property ρ.

Proof. The hypothesis states that ρ defines an E-class. Thus T1 follows from D4.

The sentential formulas therefore have the desirable properties for machine processing; namely, 1) a sentential formula consists of a finite number of machine-readable characters, and 2) a sentential formula is either elementary or of one of the designs, $\underline{\alpha}(\underline{s}_1)$ or $(\underline{s}_1)\underline{\beta}(\underline{s}_2)$.

Example: The following is a sentential formula:

$$(Ex)((B,x)AND((W,g,x)AND(W,i,x)))$$

(This says, using the interpretations of Tables 1 and 2 (p. 4), that some book was written by Russell and Whitehead.)

3.3 FREE VARIABLE SETS AND SEQUENCES

The general classification of expressions is now completed. We turn to the more specific matters that will finally lead to the definition of value set.

A central notion in the theory to follow is that of the free variables of an expression. We begin by defining this notion for basic sequences.

The free variable set of a b is the set of variables that occur in it. When this set is represented as a basic sequence whose serial order matches the first occurrences of the variables of the original (scanning the b from left to right), the result is called the free variable sequence of the expression. The number of variables in the free variable set (or the length of the free variable sequence) is called the degree of the expression.

The operation of forming free variable sequences will be denoted by 'φ'. Thus, we think of this as a machine operation that acts on an expression to produce an expression. So that φ can also operate on b's of degree zero, we define the result in that case to be Δ.

Examples: φ('x') = 'x'; φ('x,a,y,x') = 'x,y'; φ('a,b') = Δ.

When we wish to refer to the free variable set (as opposed to the sequence), we write '$\bar{\varphi}$( )'.

Examples: $\bar{\varphi}$('x,a,y,x') = {'x', 'y'}; $\bar{\varphi}$('a,b') is the null set.

It is necessary to specify another operation dealing with free variable sequences. The inputs are two b's, $\underline{b}_1$ and $\underline{b}_2$, where the second is its own free variable sequence. We write '$\delta(\underline{b}_1;\underline{b}_2)$' to indicate the deletion of all the variables of $\underline{b}_2$ from $\varphi(\underline{b}_1)$ if they occur (together with any commas if necessary). If no variables survive, we again define the result to be Δ.

Example: δ('W,x,y';'x') = 'y'.

We now extend the calculation of free variable sequences to include all sentential formulas.

D5. The rules for free variables are:

a. For elementary formulas

1) if of the form $\underline{f}^1$ (i.e., the expression 'T'),

$$\varphi(\underline{f}^1) = \Delta$$

2) if of the form $\underline{f}^n$, $n > 1$, $\varphi(\underline{f}^n)$ is as defined for basic sequences.

b. For formulas of the form $\underline{\alpha}(\underline{s})$

$$\varphi(\text{N}\emptyset\text{T}(\underline{s}_1)) = \varphi(\underline{s}_1)$$

$$\varphi((\text{E}\underline{v}_1)(\underline{s}_1)) = \delta(\varphi(\underline{s}_1);\underline{v}_1)$$

$$\varphi((\text{A}\underline{v}_1)(\underline{s}_1)) = \delta(\varphi(\underline{s}_1);\underline{v}_1)$$

$$\varphi((\text{P}\underline{b}_1^n)(\underline{s}_1)) = \varphi(\underline{b}_1^n,\varphi(\underline{s}_1))$$

(In applications of the permutation operator our main interest is in the case where the free variable set of the $\underline{b}_1^n$ is the same as that of the component.)

c. For formulas of the form $(\underline{s}_1)\underline{\beta}(\underline{s}_2)$

$$\varphi((\underline{s}_1)\underline{\beta}(\underline{s}_2)) = \varphi(\varphi(\underline{s}_1),\varphi(\underline{s}_2))$$

The degree of a formula is again defined as the number of variables in its free variable set.

Examples:

$\varphi(\text{'T'}) = \Delta$

$\varphi(\text{'(Ex)(W,x,y)'}) = \delta(\text{'x,y';'x'}) = \text{'y'}$

$\varphi(\text{'(Py,x)(W,x,y)'}) = \varphi(\text{'y,x,x,y'}) = \text{'y,x'}$

$\varphi(\text{'(W,x,y)AND(B,x)'}) = \varphi(\text{'x,y,x'}) = \text{'x,y'}$

3.4 SUBSTITUTION SCHEMATA

We now introduce the operation of substitution. The notation

$$[\underline{v}_1 \rightarrow \underline{d}_1](\underline{b}_1)$$

is used to denote the expression obtained from the basic sequence $\underline{b}_1$ by replacing every occurrence (if any) of the variable $\underline{v}_1$ by the descriptive name $\underline{d}_1$. The result of a succession of such simple substitutions is denoted by

$$[\underline{b}_1^n \rightarrow \underline{b}_2^n](\underline{b}_3)$$

where $\underline{b}_1^n$ is a basic sequence of length n and degree n (thus, $\underline{b}_1^n$ is its own free variable sequence), and $\underline{b}_2^n$ is a basic sequence of length n and degree zero (thus, $\underline{b}_2^n$ contains only descriptive names). That is, if $\underline{v}_i$ occurs in the i^{th} position of $\underline{b}_1^n$ and $\underline{d}_i$ occurs in the i^{th} position of $\underline{b}_2^n$, then the i^{th} substitution is $[\underline{v}_i \rightarrow \underline{d}_i]$. The operation name '$[\underline{b}_1^n \rightarrow \underline{b}_2^n]$' is called a <u>substitution schema</u>.

In the following discussions, we sometimes abbreviate the cumbersome but exact notation '$[\underline{v}_i \rightarrow \underline{d}_i]$' by '$\sigma_i$'.

The extension of the operation of substitution to any sentential formula is as follows:

D6. The rules for substitution in a formula $\underline{s}$ are:

a. If $\underline{v}_1$ does not occur in $\varphi(\underline{s})$, then

$$\sigma_1(\underline{s}) = \underline{s}$$

b. If $\underline{v}_1$ occurs in $\varphi(\underline{s})$, then:

1) if $\underline{s}$ is elementary (and hence a $\underline{b}$), the substitution is as defined for $\underline{b}$'s

2) if $\underline{s}$ has the form $\underline{\alpha}(\underline{s}_1)$, where $\underline{\alpha}$ is not the permutation operator, then

$$\sigma_1(\underline{\alpha}(\underline{s}_1)) = \underline{\alpha}(\sigma_1(\underline{s}_1))$$

3) if $\underline{s}$ has the form $(\underline{s}_1)\underline{\beta}(\underline{s}_2)$, then

$$\sigma_1((\underline{s}_1)\underline{\beta}(\underline{s}_2)) = (\sigma_1(\underline{s}_1))\underline{\beta}(\sigma_1(\underline{s}_2))$$

4) if $\underline{s}$ has the form $(P\underline{b}_1^n)(\underline{s}_1)$, then two situations† arise:

†A single formulation cannot be used because the design PΔ must be avoided.

a) if $\underline{b}_1^n$ is just $\underline{v}_1$ (i.e., the operator is $(P\underline{v}_1)$), then

$$\sigma_1((P\underline{b}_1^n)(\underline{s}_1)) = \sigma_1(\underline{s}_1)$$

b) if $\underline{b}_1^n$ is not $\underline{v}_1$, then

$$\sigma_1((P\underline{b}_1^n)(\underline{s}_1)) = (P\delta(\underline{b}_1^n;\underline{v}_1))(\sigma_1(\underline{s}_1))$$

Example: ['x' → 'a']('(B,x)AND((Ex)(W,y,x))') = '(B,a)AND((Ex)(W,y,x))'

By using the inductive principle (T1), we can show various things concerning substitution and free variable sequences. The results are given in the following theorem:†

T2. Concerning substitution and free variable calculations we have:

a. Substitution is defined for all $\underline{s}$.

b. Substitution in a sentential formula yields a sentential formula. (This is because of our weak definition of elementary formula (D2).)

c. Substitutions are commutative. (For example, $\sigma_i(\sigma_j(\underline{s})) = \sigma_j(\sigma_i(\underline{s}))$.)

d. $\wp$ is defined for all $\underline{s}$.

e. $\wp([\underline{v}_1 \rightarrow \underline{d}_1](\underline{s})) = \delta(\wp(\underline{s});\underline{v}_1)$.

---

†Proofs omitted.

3.5 VALUE SETS

The File. The logical preliminaries necessary to define the notion of value set are complete. The only remaining prerequisite is a data base. This data base is a set of elementary sentential formulas of degree zero. This distinguished set is called the file. The motivation is that the file shall consist of our stock of "true" sentences.

It is convenient to think of the file as a matrix whose rows correspond to the elementary sentential formulas. If we think of the elements as computer words (as we did for basic sequences), then the file becomes a "dimensioned array" (in the FORTRAN sense). The differing lengths can be treated by filling out the row with a special word.

An example file is given in Table 2 (p. 4). In the present discussion, however, since we do not use the "identifier" column, we can regard the first column as deleted.

Value Sets of Elementary Sentential Formulas. We begin with the definition of value set for the elementary formulas. Since the generation of value sets will be a machine routine, we assign a special designation 'ω' to this operation. The first step is to define $\omega(\underline{f})$ for every $\underline{f}$ of degree zero.

What value should we assign to such a formula? We would like the value to correspond somehow to the "truth value." Several approaches are possible.

One method is to assume that the file gives an exhaustive description of the state of our universe of discourse. That is, the elementary formulas in the file

tell us exactly which of the given individuals have the given properties, and which have the given relations between them. The value of an elementary formula is then "true" if the formula is in the file, and "false" otherwise.

We can think of this first method as a kind of inference; we infer the truth value of elementary formulas not in the file on the basis of a rather strong assumption about the structure of the file.

A second method would involve inferences based on more complicated criteria. One set of criteria would again concern the structure of the file, but only in regard to certain expressions. For example, the formula 'W,x,b' (see Table 2, p. 4) could be regarded as complete in the sense that all of its true instances (obtained by substitutions for 'x') appear in the file. Hence, we can infer that 'W,e,b' is false because it does not appear in the file. The second set of criteria would be meaning postulates [4], which reflect our understanding of the sense of certain descriptive names. For example, we might have a postulate stating that a book is not a paper. The symbolic form of this is

$$(B,x)\,IMP(N\emptyset T(P,x))$$

Thus, from the file entry

B,b

(Table 2, p. 4), we can infer that 'P,b' is false.

A third method would involve quantitative implementation of the notions of plausible inference discussed in

Ref. 1. This approach, technically more difficult than the others, would require the use of probabilistic truth tables. Here, instead of truth values we would use a quantitative "logic of weights" [5] or the "degree of confirmation" [6]. (The first method can be regarded as a gross approximation to such a logic of weights† if we assign a rather high weight (near one) to an elementary sentence in the file and a rather low weight (near zero) to those not in the file.)

For the present study we use the first method described above. This approach is admittedly crude, but the results are of both practical and theoretical interest.

Based on the above considerations, we define:

D7. The value set of an elementary formula is given by the following rules:

a. If $\underline{f}$ is of degree zero, then

1) for $\underline{f}^1$ (i.e., the expression 'T')

$$\omega(\underline{f}^1) = 1$$

2) for $\underline{f}^n$, $n > 1$,

$\omega(\underline{f}^n) = 1$, if $\underline{f}^n$ is in the file

$\omega(\underline{f}^n) = 0$, otherwise.

---

†The details of the reduction of two-valued logic to a logic of weights are explained in Ref. 5, pp. 398-402.

b. If $\underline{f}$ is of degree m, m > 0, then

$$\omega(\underline{f}) = \{\underline{b}^m \mid \omega([\varphi(\underline{f}) \rightarrow \underline{b}^m](\underline{f})) = 1\}$$

This definition specifies $\omega(\underline{f})$ exactly, but let us make it more concrete by considering a possible representation of the resulting data. Suppose $\omega(\underline{f})$ is to be stored in an array of m columns, each column being labelled by the free variable sequence of $\underline{f}$. Each row then represents a member of $\omega(\underline{f})$. So that formulas of degree zero can also be represented by arrays, let us represent $\omega(\underline{f})$, where $\underline{f}$ is of degree zero, as a 2x1 array with the two entries being Δ and $\omega(\underline{f})$.

Examples: ω('W,x,h') = {'g', 'i'}, ω('W,g,h') = 1. In terms of an array representation, these two results are represented by

x
g
i

and

1

respectively.

The General Definition of Value Set. We now extend the concept of value set to any sentential formula. The necessary definitions are given here, followed by the proof that they are well-formed. The problem of actually computing value sets will be taken up in Sec. 4 below.

The definition of value set is broken into three parts (D8, D9, D10), which correspond to the cases:
1) formulas of degree zero with components of degree zero;
2) formulas of degree zero with components of degree one;
3) formulas of degree greater than zero.

D8. If $\underline{s}$ is of degree zero and of the form $N\emptyset T(\underline{s}_1)$ or $(\underline{s}_1)\underline{\beta}(\underline{s}_2)$, then $\omega(\underline{s})$ is given by the following arithmetic rules:†

a. For $N\emptyset T(\underline{s}_1)$,

$$\omega(N\emptyset T(\underline{s}_1)) = 1 - \omega(\underline{s}_1)$$

b. For $(\underline{s}_1)\underline{\beta}(\underline{s}_2)$, let q be the quantity $|\omega(\underline{s}_1) - \omega(\underline{s}_2)|$, then

$$\omega((\underline{s}_1)AND(\underline{s}_2)) = \tfrac{1}{2}[\omega(\underline{s}_1) + \omega(\underline{s}_2) - q]$$

$$\omega((\underline{s}_1)\emptyset R(\underline{s}_2)) = \tfrac{1}{2}[\omega(\underline{s}_1) + \omega(\underline{s}_2) + q]$$

$$\omega((\underline{s}_1)BN(\underline{s}_2)) = \tfrac{1}{2}[\omega(\underline{s}_1) - \omega(\underline{s}_2) + q]$$

$$\omega((\underline{s}_1)IMP(\underline{s}_2)) = 1 - \tfrac{1}{2}[\omega(\underline{s}_1) - \omega(\underline{s}_2) + q]$$

$$\omega((\underline{s}_1)IFF(\underline{s}_2)) = 1 - q$$

---

†These are equivalent to the truth tables of the sentential calculus.

D9. If $\underline{s}$ is of degree zero and of the form $(E\underline{v})(\underline{s}_1)$ or $(A\underline{v})(\underline{s}_1)$, then $\omega(\underline{s})$ is given by the rules:

a. If $\underline{s}_1$ is of degree zero, then

$$\omega((E\underline{v})(\underline{s}_1)) = \omega(\underline{s}_1)$$

$$\omega((A\underline{v})(\underline{s}_1)) = \omega(\underline{s}_1)$$

b. If $\underline{s}_1$ is of degree one, then

$\omega((E\underline{v})(\underline{s}_1)) = 0$, if $\omega(\underline{s}_1)$ is null

$\omega((E\underline{v})(\underline{s}_1)) = 1$, otherwise

$\omega((A\underline{v})(\underline{s}_1)) = 0$, if there is a $\underline{d}$ not in $\omega(\underline{s}_1)$

$\omega((A\underline{v})(\underline{s}_1)) = 1$, otherwise.

D10. If s is of degree $m > 0$, then

$$\omega(\underline{s}) = \{\underline{b}^m \mid \omega([\sigma(\underline{s}) \rightarrow \underline{b}^m](\underline{s})) = 1\}$$

Note that all cases are treated. This can be shown by D5, for if a formula is of degree zero, then: 1) it cannot be of the form $(P\underline{b})(\underline{s})$; 2) if it is of the form $(E\underline{v})(\underline{s})$ or $(A\underline{v})(\underline{s})$, its component must be of degree zero or degree one.

We now show that the definitions are well-formed. First we discuss a key point: the consistency of D10. Introduce the following notation: Let $\underline{s}$ be of degree m with

$$\omega(\underline{s}) = \underline{v}_1, \underline{v}_2, \ldots, \underline{v}_m \tag{1}$$

Consider now a $\underline{b}^m$, say

$$\underline{b}^m = \underline{d}_1, \underline{d}_2, \ldots, \underline{d}_m \tag{2}$$

Select two subsequences from $\omega(\underline{s})$ taken in the same linear order; the first consisting of p variables

$$\underline{v}_{i_1}, \underline{v}_{i_2}, \ldots, \underline{v}_{i_p} \tag{3}$$

and the second, the remaining q = m - p variables

$$\underline{v}_{j_1}, \underline{v}_{j_2}, \ldots, \underline{v}_{j_q} \tag{4}$$

Define by the indices so determined the corresponding $\underline{b}$'s

$$\underline{b}_i^p = \underline{d}_{i_1}, \underline{d}_{i_2}, \ldots, \underline{d}_{i_p} \tag{5}$$

and

$$\underline{b}_j^q = \underline{d}_{j_1}, \underline{d}_{j_2}, \ldots, \underline{d}_{j_q} \tag{6}$$

Now let $\underline{s}_i$ be the following substitution instance of $\underline{s}$

$$\underline{s}_i = \sigma_{j_1}(\sigma_{j_2}(\ldots(\sigma_{j_q}(\underline{s}))\ldots)) \tag{7}$$

i.e., the schema substitutes (6) for (4). We have

$$\sigma(\underline{s}_i) = \underline{v}_{i_1}, \underline{v}_{i_2}, \ldots, \underline{v}_{i_p} \tag{8}$$

It then follows by commutativity of substitutions that the formulations

$$[\sigma(\underline{s}_i) \rightarrow \underline{b}_i^p](\underline{s}_i) \tag{9}$$

and

$$[\sigma(\underline{s}) \rightarrow \underline{b}^m](\underline{s}) \tag{10}$$

are descriptions of the same formula (of degree zero). Consequently, for D10 to be consistent we must have

$$\underline{b}^m \in \omega(\underline{s}) \text{ if and only if } \underline{b}_i^p \in \omega(\underline{s}_i) \tag{11}$$

(For a simple substitution, (11) takes the following form: if $\underline{s}$ is of degree $m > 1$ and $\underline{v}_i$ occurs in the i^{th} position of $\sigma(\underline{s})$, then

$$\underline{d}_1, \ldots, \underline{d}_i, \ldots, \underline{d}_m \in \omega(\underline{s})$$

if and only if

$$\underline{d}_1, \ldots, \underline{d}_{i-1}, \underline{d}_{i+1}, \ldots, \underline{d}_m \in \omega([\underline{v}_i \rightarrow \underline{d}_i](\underline{s})) \tag{12}$$

On the other hand, if $m = 1$ (so that $\underline{v}_1 = \sigma(\underline{s})$), then we have

$$\underline{d}_1 \in \omega(\underline{s}) \text{ if and only if } \omega([\underline{v}_1 \rightarrow \underline{d}_1](\underline{s})) = 1 \tag{13}$$

which is D10 with m = 1.)

We now prove the following theorem:

T3. For all sentential formulas $\underline{s}$

a. $\omega(\underline{s})$ is defined

b. If $\underline{s}$ is of degree $m > 1$, then for all substitution instances $\underline{s}_i$ of $\underline{s}$,

$$\underline{b}^m \in \omega(\underline{s}) \text{ if and only if } \underline{b}_i^p \in \omega(\underline{s}_i)$$

(where $\underline{b}_i^p$ and $\underline{s}_i$ are as defined by (5) and (7) above).

Proof. We consider T3 as a property of formulas and use the inductive principle (T1). First, we establish a preliminary result to facilitate the proof. (In what follows we use '$\underline{r}$', '$\underline{s}$', '$\underline{t}$' to indicate sentential formulas. Thus substitution instances of formulas can be indicated by the use of subscripts. $\underline{t}$ will be the formula under consideration, and $\underline{r}$ and $\underline{s}$ will be the components of $\underline{t}$.)

I. If T3a has been proved for $\underline{t}$ and $\underline{t}_i$ (where $\underline{t}_i$ is a substitution instance as defined by (7) above), then we can assert T3b.

For, if T3a holds for $\underline{t}$, we can transform (D10)

$$\underline{b}^m \in \omega(t)$$

into

$$\omega([\varphi(\underline{t}) \rightarrow \underline{b}^m](\underline{t})) = 1$$

But by our remarks on (9) and (10), this is the same as

$$\omega([\varphi(\underline{t}_i) \rightarrow \underline{b}_i^p](\underline{t}_i)) = 1$$

To take the final step and transform this by D10, we need only know that $\omega(\underline{t}_i)$ has been defined. Hence, by hypothesis,

$$\underline{b}_i^p \in \omega(\underline{t}_i)$$

II. T3 holds for elementary formulas.

A. $\omega(\underline{f})$ is explicitly given for $\underline{f}$ of degree zero (D7a).

B. If $\underline{f}$ is of degree $m > 0$, then any full substitution $[\varphi(\underline{f}) \rightarrow \underline{b}^m]$ in $\underline{f}$ gives an elementary formula of degree zero; hence $\omega(\underline{f})$ is determined by D7b. (D7b is, of course, the same as D10.)

C. The substitution instance $\underline{f}_i$ of $\underline{f}$ is elementary. Thus, by part B, $\omega(\underline{f}_i)$ is defined. T3b now follows by I.

We continue the proof by next treating all cases of degree zero.

III. If T3 holds for $\underline{r}$ and $\underline{s}$, and $\underline{t}$ is of degree zero and is of one of the forms $\underline{\alpha}(\underline{r})$ or $(\underline{r})\underline{\beta}(\underline{s})$, then ω is defined for $\underline{t}$.

By the remark following D10, all cases of degree zero are covered by D8 and D9. But these explicitly give $\omega(\underline{t})$ in terms of $\omega(\underline{r})$ and $\omega(\underline{s})$.

It remains only to show (as required by D10) that if $\underline{t}$ is not elementary and of degree $m > 0$, then

$\omega([\varphi(\underline{t}) \rightarrow \underline{b}^m](\underline{t}))$ can be expressed in terms of the value sets of the components of $\underline{t}$. This will then prove T3a. If we show the same result for $\underline{t}_i$, a substitution instance of $\underline{t}$, then by I, T3b will be proved.

IV. If T3 holds for $\underline{r}$ and $\underline{s}$, and $\underline{t}$ is of degree $m > 0$ and of the form $\underline{\alpha}(\underline{r})$ or $(\underline{r})\underline{\beta}(\underline{s})$, then T3 holds for $\underline{t}$.

A. By hypothesis, T3 holds for any substitution instances of $\underline{r}$ and $\underline{s}$.

B. For any $\underline{\alpha}$, except the permutation operator, and any $\underline{\beta}$, the substitution instance $[\varphi(\underline{t}) \rightarrow \underline{b}^m](\underline{t})$ describes, by D6, a formula with the same operator, but whose components are substitution instances of the original components. If $\underline{\alpha}$ is the permutation operator, then, by D6, the operator is eliminated and the formula becomes a substitution instance of the component. Hence, in every case,

$$\omega([\varphi(\underline{t}) \rightarrow \underline{b}^m](\underline{t}))$$

is determined by part A and III. This shows T3a.

C. To show T3b, note that the above argument is valid for partial substitutions. That is, let $\underline{t}_i$ be a partial substitution instance of $\underline{t}$, and let $\underline{r}_j$ and $\underline{s}_k$ be the subsequently induced partial substitution instances of $\underline{r}$ and $\underline{s}$. By part A, we can then apply part B to $\underline{t}_i$ with components $\underline{r}_j$ and $\underline{s}_k$, and assert that T3a holds for $\underline{t}_i$. T3b now follows from I.

Combining II, III, and IV with the inductive principle (T1) completes the proof.

We conclude this section with some additional results. The first relates the value set of $(E\underline{v})(\underline{s})$ to the value set of $\underline{s}$.

T4. If $\underline{s}$ is of degree $m > 1$ and $\underline{v}_i$ occurs in the i^{th} position of $\varphi(\underline{s})$ and $\underline{b}_o^{m-1}$ is any sequence of $\underline{d}$'s

$$\underline{b}_o^{m-1} = \underline{d}_1, \ldots, \underline{d}_{i-1}, \underline{d}_{i+1}, \ldots, \underline{d}_m$$

then:

$$\underline{b}_o^{m-1} \in \omega((E\underline{v}_i)(\underline{s}))$$

if and only if there is a $\underline{d}_i$ such that

$$\underline{d}_1, \ldots, \underline{d}_i, \ldots, \underline{d}_m \in \omega(\underline{s})$$

(i.e., $\underline{d}_1, \ldots, \underline{d}_i, \ldots, \underline{d}_m$ is formed from $\underline{b}_o^{m-1}$ by the appropriate insertion of $\underline{d}_i$).

Proof.

$$\underline{b}_o^{m-1} \in \omega((E\underline{v}_i)(\underline{s}))$$

is transformed by D10, D6, and D9 into: there is a $\underline{d}_i$ such that

$$\underline{d}_i \in \omega([\varphi((E\underline{v}_i)(\underline{s})) \rightarrow \underline{b}_o^{m-1}](\underline{s}))$$

This in turn can be transformed by D10 and D5 into: there is a $\underline{d}_i$ such that

$$\omega([o(\underline{s}) \rightarrow \underline{b}_o^m](\underline{s})) = 1$$

where $\underline{b}_o^m$ is $\underline{d}_1, \ldots, \underline{d}_i, \ldots, \underline{d}_m$. Since these transformations are reversible, this proves T4.

The second result gives some examples of formulas that are co-valued.† (Proofs are omitted.)

T5.

a. $\omega(N\emptyset T(\underline{s})) = \omega((T)BN(\underline{s}))$

b. $\omega((\underline{r})AND(\underline{s})) = \omega((\underline{r})BN(N\emptyset T(\underline{s})))$

c. $\omega((\underline{r})\emptyset R(\underline{s})) = \omega(N\emptyset T((N\emptyset T(\underline{r}))BN(\underline{s})))$

d. $\omega((\underline{r})IMP(\underline{s})) = \omega(N\emptyset T((\underline{r})BN(\underline{s})))$

e. $\omega((\underline{r})IFF(\underline{s})) = \omega(((\underline{r})IMP(\underline{s}))AND((\underline{s})IMP(\underline{r})))$

f. $\omega((A\underline{v})(\underline{s})) = \omega(N\emptyset T((E\underline{v})(N\emptyset T(\underline{s}))))$

Formulas such as these will be used in Sec. 5 below.

Examples: Let $\underline{r}$ be the formula

$$(B,y)AND(W,x,y)$$

Thus, the free variable sequence of $\underline{r}$ is 'y,x'. Representing $\omega(\underline{r})$ as an array, we get by D10 and our example file (Table 2, p. 4)

---

†If two formulas are logically equivalent when interpreted as formulas of the predicate calculus and their free variable sequences are the same, then they have the same value set. See Sec. 5.2 below for additional discussion.

y	x
b	a
d	a
f	e
h	g
h	i

Consider now the formula s (with r as component)

$$(Ey)((B,y)AND(W,x,y))$$

We can use T4 to evaluate $\omega(s)$. The result is given by

x
a
e
g
i

(The formula s corresponds to the question 'Who has written a book?'. $\omega(s)$ can be considered as giving the answer to that question.)

3.6 A NOTE ON THE PERMUTATION OPERATOR

Having defined 'free variable sequence' and 'value set', we can now describe the use of the permutation operator.

We see from D5 (p. 26) that the permutation operator can be used to change the ordering of the free variable sequence of a formula; for our purposes, this is its primary use. That is, we use the permutation operator to specify in which order the machine is to process certain information. For example, suppose we want the machine to print the value set of 'x was written by y', but the file does not contain the particular relation was written by. We then form the sentential formula 'W,y,x'. Since the free variable sequence of this is 'y,x', the listings of the value set are pairs of names with the first members corresponding to 'y' and the second to 'x'. To change this ordering, we prefix 'W,y,x' with the permutation operator '(Px,y)'.

The permutation operator is similar, in some respects, to the λ-operator[†] [7] used to define a relation in terms of a complex predicate expression. Thus, in the above discussion, we have essentially defined the converse of W.

By also allowing the permutation operator to contain variables not in the component, we obtain a freedom of expression similar to that given by the λ-operator. For example, a formulation of the universal relation is stated by means of the sentential formula '(Px,y)(T)'. Although such formulas are not of particular interest from a standpoint of machine processing (except to prohibit them[††]), it is of theoretical interest to include them in the class of sentential formulas. The price we pay is a slight complication of our rules (e.g., D6b-4, p. 28).

[†] See also Sec. 4.7 (pp. 81-83).

[††] See T22, Sec. 4.4 (p. 64), on the definitude of (Pb)(s).

4. THE THEORY OF PROPER FORMULAS

4.1 INITIAL REMARKS

In the preliminary discussion of Sec. 2, we claimed that certain difficulties arising in the calculation of value sets could be avoided by investigating a special class of formulas, called proper formulas. In this section we develop this theory.

The first notion we need is that of definite formula. This notion is intended to give a precise characterization of what constitutes a "reasonable" question. That is, for us a "reasonable" question will be one whose symbolic question is definite.

The second step is the definition of proper formula. This rests on the definition of definite formula. Exactly: a formula is proper if 1) it is elementary, or 2) it is definite and its components are proper. It turns out that elementary formulas are definite. Thus, proper formulas are definite in a very strong sense.

The third step is to characterize formulas with definite components. The most desirable result here is one in which the characterization is machine-recognizable and a theorem of the following kind is obtained: If s is a formula with definite components, then s is definite if and only if the characterization holds. By definition of proper formula, it will then follow, as a corollary, that the same theorem holds if 'definite' is replaced throughout by 'proper'. It will also follow, by definition of proper formula, that the characterization is a necessary condition for propriety. (The characterization

is not sufficient without additional information on the components.)

Various theories are obtained by using different definitions of definite formula. One such theory, presented below, is based on the criterion of definitude suggested in Sec. 2, namely, that a definite formula is one whose value set is invariant under the addition of a new descriptive name to the dictionary. We now develop this idea formally.

4.2 DEFINITE FORMULAS AND PROPRIETY

We define definite formula, proper formula, and state some preliminary results.

The notion of definite formula is developed, first, relative to a given data base. This restricted notion is called semi-definite.

By a data base we mean a dictionary† of descriptive names, d, together with a file. We stipulate that a special symbol, '*', is excluded from our list of elements and hence from the dictionary (see Sec. 3.2). By the definitions of Sec. 3, '*' does not then occur in any sentential formula or the file.

D11a. Given a data base $\mathfrak{D}$ and a formula s on $\mathfrak{D}$, we define s to be semi-definite with respect to $\mathfrak{D}$ by means of the following logical test procedure:

1) Calculate $\omega(s)$.

†The dictionary, considered as consisting of expressions of an object language, inventories the individuals of the universe of discourse as well as the primitive properties and relations of this universe.

2) Form a "pseudo" data base $\mathcal{A}_*$ by adding '*' to the dictionary of $\mathcal{A}$. (This, of course, leaves the file unchanged.)

3) Calculate the value set of $\underline{s}$ on $\mathcal{A}_*$. Denote the result by '$\omega_*(\underline{s})$'. (Thus, the calculation of $\omega_*(\underline{s})$ on $\mathcal{A}_*$ allows the new substitution $[v_i \rightarrow *]$.)

4) If $\omega_*(\underline{s}) = \omega(\underline{s})$, then we say that $\underline{s}$ is semi-definite on $\mathcal{A}$.

Note that if $\underline{s}$ is of degree $m > 0$, then both $\omega(\underline{s})$ and $\omega_*(\underline{s})$ are sets of basic sequences of length m. Consequently, $\underline{s}$ can never be semi-definite on $\mathcal{A}$ if $\omega_*(\underline{s})$ contains a $\underline{b}^m$ in which the element '*' occurs.

Definite formula is now defined as follows.

D11b. $\underline{s}$ is definite if and only if $\underline{s}$ is semi-definite on every data base.

Thus, the definitude of a formula is a property of only the formula and is independent of the contents of the file.

Example: '(B,c)BN(W,x,c)' is semi-definite with respect to the example data base (Tables 1 and 2, p. 4), but is not definite. This is because

$$\omega(\text{'B,c'}) = 0$$

Consequently, no substitution for 'x' makes the formula true. Hence,

$$\omega_*(\underline{s}) = \omega(\underline{s}) = \text{the null set}$$

If, however, we change the file so that

$$\omega('B,c') = 1$$

then

$$\omega_*(\underline{s}) = \omega(\underline{s}) \cup \{'*'\}$$

T6. If $\underline{s}$ is elementary, then $\underline{s}$ is definite.

Proof. Let $\underline{s}$ be elementary. If $\underline{s}$ is of degree zero, then, since the files of $\mathfrak{A}$ and $\mathfrak{A}_*$ are the same, $\omega(\underline{s}) = \omega_*(\underline{s})$. If $\underline{s}$ is of degree $m > 0$, then the substitution instances of $\underline{s}$ having value 1 on $\mathfrak{A}$ are exactly the substitution instances of $\underline{s}$ having value 1 on $\mathfrak{A}_*$. This proves T6.

D12. $\underline{s}$ is proper if and only if

1) $\underline{s}$ is elementary (hence definite by T6),

or

2) $\underline{s}$ is definite and its components are proper.

By T6 and D12 we have

T7. If $\underline{s}$ is proper, then $\underline{s}$ is definite.

The next two theorems concern substitution. First, substitution in a definite formula yields a definite formula.

T8. If $\underline{s}$ is definite, then $[\underline{v}_1 \rightarrow \underline{d}_1](\underline{s})$ is definite.

Proof. Let $\underline{v}_1 \in \bar{\sigma}(\underline{s})$ (otherwise the result is trivial). If $[\underline{v}_1 \rightarrow \underline{d}_1](\underline{s})$ is of degree zero, then

$$\omega_*([\underline{v}_1 \rightarrow \underline{d}_1](\underline{s})) = 1$$

is equivalent by D10 to

$$\underline{d}_1 \in \omega_*(s)$$

Hence, by hypothesis, this is equivalent to

$$\underline{d}_1 \in \omega(\underline{s})$$

and, by D10 again, to

$$\omega([\underline{v}_1 \rightarrow \underline{d}_1](\underline{s})) = 1$$

Now let $[\underline{v}_1 \rightarrow \underline{d}_1](\underline{s})$ be of degree greater than zero. Then $\underline{s}$ is of degree $m > 1$. Since $\underline{s}$ is definite by hypothesis, we have (D11)

$$\underline{b}^m \in \omega(\underline{s}) \text{ if and only if } \underline{b}^m \in \omega_*(\underline{s}) \qquad (1)$$

Let $\underline{s}_i$ be $[\underline{v}_1 \rightarrow \underline{d}_1](\underline{s})$. Now apply T3b to both ω and ω_*. We get

$$\underline{b}^m \in \omega(\underline{s}) \text{ if and only if } \underline{b}_i^{m-1} \in \omega(\underline{s}_i) \qquad (2)$$

and

$$\underline{b}^m \in \omega_*(\underline{s}) \text{ if and only if } \underline{b}_i^{m-1} \in \omega_*(\underline{s}_i) \qquad (3)$$

By (1), (2), and (3) we have

$$\underline{b}_i^{m-1} \in \omega_*(\underline{s}_i) \text{ if and only if } \underline{b}_i^{m-1} \in \omega(\underline{s}_i)$$

Hence $\omega_*(\underline{s}_i) = \omega(\underline{s}_i)$, and so $\underline{s}_i$ is definite. This proves T8.

The analogous result holds for proper formulas:

T9. If $\underline{s}$ is proper, then $[\underline{v}_1 \rightarrow \underline{d}_1](\underline{s})$ is proper.

Proof. We show this by the inductive principle (T1). Consider the statement of T9 to give a property of $\underline{s}$. Now if $\underline{s}$ is elementary, then so is a substitution instance of $\underline{s}$; hence T9 holds by D12. Next, let T9 hold for $\underline{r}$ and let $\underline{t}$ be $\underline{\alpha}(\underline{r})$. If $\underline{t}$ is proper, then by D12 $\underline{t}$ is definite and $\underline{r}$ is proper. Applying T8 to $\underline{t}$, we see that $[\underline{v}_1 \rightarrow \underline{d}_1](\underline{t})$ is definite. Now, by D6, $[\underline{v}_1 \rightarrow \underline{d}_1](\underline{t})$ is the same as $\underline{\alpha}([\underline{v}_1 \rightarrow \underline{d}_1](\underline{r}))$. Therefore, by the hypothesis that T9 holds for $\underline{r}$, $[\underline{v}_1 \rightarrow \underline{d}_1](\underline{t})$ has a proper component. Consequently, $[\underline{v}_1 \rightarrow \underline{d}_1](\underline{t})$ is proper (D12). A similar argument establishes the result for the form $(\underline{r})\underline{\beta}(\underline{s})$. Thus, by T1, T9 holds for all s. This proves T9.

4.3 CERTAIN FORMULAS OF DEGREE ZERO

It is convenient at this point to deal with certain special cases. These are formulas of degree zero having an operator that does not alter the free variable set of a formula.

T10. If $\underline{t}$ is of degree zero and of either the form $N\emptyset T(\underline{r})$ or the form $(\underline{r})\underline{\beta}(\underline{s})$, and $\underline{r}$ and $\underline{s}$ are definite, then $\underline{t}$ is definite.

Proof. Since $\underline{t}$ is of degree zero, then by D5, so are $\underline{r}$ and $\underline{s}$. By D8, $\omega_*(\underline{t})$ is an arithmetic function of $\omega_*(\underline{r})$ and $\omega_*(\underline{s})$. By the hypothesis that both $\underline{r}$ and $\underline{s}$ are definite, it then follows from D8 that $\omega_*(\underline{t}) = \omega(\underline{t})$. This proves T10.

As a corollary to T10 we have by D12 and T7:

T11. If $\underline{t}$ is of degree zero and of either the form $N\emptyset T(\underline{r})$ or the form $(\underline{r})\beta(\underline{s})$, and $\underline{r}$ and $\underline{s}$ are proper, then $\underline{t}$ is proper.

It now follows that any formula of the "propositional" calculus (i.e., a formula of degree zero made up of only the operators of class β and $N\emptyset T$) is proper. This is because the following hold: 1) elementary formulas are proper (T6, D12); 2) if $\underline{r}$ and $\underline{s}$ are proper of degree zero, then $N\emptyset T(\underline{r})$ and $(\underline{r})\beta(\underline{s})$ are proper of degree zero (T11). Hence, by the inductive principle (T1), we have:

T12. Any formula of degree zero that is either elementary or made up entirely of the operators of class β or $N\emptyset T$ is proper.

4.4 FORMULAS WITH DEFINITE COMPONENTS

In this section a systematic study is made of formulas with definite components.

T13. If $\underline{s}$ is definite, then:

$N\emptyset T(\underline{s})$ is definite if and only if $\bar{\varphi}(\underline{s})$ is null.

Proof. A. To establish the implication from left to right, we show the contraposit. Let t be $N\emptyset T(\underline{s})$. Suppose $\bar{\varphi}(\underline{s})$ is not null. Then $\underline{s}$ is of degree $m > 0$ and so is $\underline{t}$ (D5). By hypothesis

$$\omega_*(\underline{s}) = \omega(\underline{s}) \qquad (4)$$

Now form a $\underline{b}^m$ that contains '*' as an element. By definition, $\underline{b}^m \notin \omega(\underline{s})$; hence, by (4), $\underline{b}^m \notin \omega_*(\underline{s})$. Thus by D10

$$\omega_*([\varphi(\underline{s}) \rightarrow \underline{b}^m](\underline{s})) = 0$$

and so by D8

$$\omega_*(N\emptyset T([\varphi(\underline{s}) \rightarrow \underline{b}^m](\underline{s}))) = 1 \qquad (5)$$

But, since $\varphi(\underline{s}) = \varphi(\underline{t})$, (5) can be transformed by D6 into

$$\omega_*([\varphi(\underline{t}) \rightarrow \underline{b}^m](\underline{t})) = 1 \qquad (6)$$

Therefore, by (6) and D10, we have $\underline{b}^m \in \omega_*(\underline{t})$. Hence $\underline{t}$ is indefinite.

B. For the implication from right to left, note that if $\bar{\varphi}(\underline{s})$ is null, then by D5 so is $\bar{\varphi}(\underline{t})$. Hence $\underline{t}$ is of degree zero and therefore definite by T10.

This proves T13.

We consider next the formulas of the form $(\underline{r})\beta(\underline{s})$. To facilitate the discussion, we use a notational device similar to that introduced previously (Sec. 3.5) for substitution instances. Consider the sentential formulas $\underline{t}$, $\underline{r}$, and $\underline{s}$ where

$$\underline{t} = (\underline{r})\beta(\underline{s})$$

Let

$$\varphi(\underline{t}) = \underline{v}_1, \underline{v}_2, \ldots, \underline{v}_m \qquad (7)$$

and let

$$\underline{b}^m = \underline{d}_1, \underline{d}_2, \ldots, \underline{d}_m \tag{8}$$

By D5c, $\varphi(\underline{r})$ and $\varphi(\underline{s})$ are given as certain subsequences of (7). Thus, if $\underline{r}$ and $\underline{s}$ are of degrees p and q, respectively:

$$\varphi(\underline{r}) = \underline{v}_{i_1}, \underline{v}_{i_2}, \ldots, \underline{v}_{i_p} \tag{9}$$

$$\varphi(\underline{s}) = \underline{v}_{j_1}, \underline{v}_{j_2}, \ldots, \underline{v}_{j_q} \tag{10}$$

(Possibly one of these could contain no variables, even if $m > 0$.) The indices of these subsequences can then be used to select from (8)

$$\underline{b}_i^p = \underline{d}_{i_1}, \underline{d}_{i_2}, \ldots, \underline{d}_{i_p}$$

and

$$\underline{b}_j^q = \underline{d}_{j_1}, \underline{d}_{j_2}, \ldots, \underline{d}_{j_q}$$

Hence, by D6, the substitution schema $[\varphi(\underline{t}) \rightarrow \underline{b}^m]$ when applied to $\underline{t}$ induces substitutions in $\underline{r}$ and $\underline{s}$, namely, $[\varphi(\underline{r}) \rightarrow \underline{b}_i^p]$ and $[\varphi(\underline{s}) \rightarrow \underline{b}_j^q]$. We then define $\underline{r}_i$ and $\underline{s}_j$ to be the following substitution instances of $\underline{r}$ and $\underline{s}$, respectively:

$$\underline{r}_i = [\varphi(\underline{r}) \rightarrow \underline{b}_i^p](\underline{r}) \tag{11}$$

$$\underline{s}_j = [\varphi(\underline{s}) \rightarrow \underline{b}_j^q](\underline{s}) \tag{12}$$

We next have a lemma to aid in the analysis to follow.

T14. If $\underline{t}$ is of degree $m > 0$ and of the form $(\underline{r})\underline{\beta}(\underline{s})$ with $\underline{r}$ and $\underline{s}$ definite, then

$$\omega(\underline{t}) \subset \omega_*(\underline{t})$$

(For the inclusion[†] in the other direction, the most that can be said is that for a given data base the intersection of $\omega_*(\underline{t})$ with the basic sequences of the data base is included in $\omega(\underline{t})$.)

Proof. A. Let $\underline{b}^m \in \omega(\underline{t})$. By D10, this is equivalent to

$$\omega([\varphi(\underline{t}) \rightarrow \underline{b}^m](\underline{t})) = 1$$

which is to say

$$\omega((\underline{r}_i)\underline{\beta}(\underline{s}_j)) = 1$$

Since $\underline{r}$ and $\underline{s}$ are definite, then so are $\underline{r}_i$, $\underline{s}_j$ (T8). Because $(\underline{r}_i)\underline{\beta}(\underline{s}_j)$ is of degree zero, it therefore follows, by T10, that

---

[†]The sign '$\subset$' for inclusion between sets allows also equality of sets.

$$\omega_*([\varphi(\underline{t}) \rightarrow \underline{b}^m](\underline{t})) = 1$$

and so, by D10, $\underline{b}^m \in \omega_*(\underline{t})$.

B. The argument is not reversible, because $\underline{b}^m \in \omega_*(\underline{t})$ does not generally imply that $\underline{r}_i$ and $\underline{s}_j$ are permissible formulas of a given data base (unless, of course, we stipulate that $\underline{b}^m$ does not contain '*').

This proves T14.

T15. If $\underline{r}$ and $\underline{s}$ are definite, then

$$(\underline{r})\text{AND}(\underline{s}) \text{ is definite.}$$

Proof. Let $\underline{t}$ be $(\underline{r})\text{AND}(\underline{s})$. If $\underline{t}$ is of degree zero, then the theorem follows from T10. Therefore, let $\underline{t}$ be of degree $m > 0$. By T14

$$\omega(\underline{t}) \subset \omega_*(\underline{t})$$

Hence, we need only show that

$$\omega_*(\underline{t}) \subset \omega(\underline{t}) \tag{13}$$

Suppose that $\underline{b}^m \in \omega_*(\underline{t})$, then

$$\omega_*([\varphi(\underline{t}) \rightarrow \underline{b}^m](\underline{t})) = 1$$

That is,

$$\omega_*((\underline{r}_i)\text{AND}(\underline{s}_j)) = 1$$

By D8, this means that

$$\omega_*(\underline{r}_i) = 1, \ \omega_*(\underline{s}_j) = 1$$

If both $\underline{r}_i$ and $\underline{s}_j$ are formulas of the data base, we can go on to assert

$$\omega(\underline{r}_i) = 1, \ \omega(\underline{s}_j) = 1$$

and hence, that $\underline{b}^m \in \omega(\underline{t})$. Suppose then that one of them, say $\underline{r}_i$, is not a permissible formula. Then $\omega_*(\underline{r}_i) = 1$ means that $\underline{b}_i^p \in \omega_*(\underline{r})$, where $\underline{b}_i^p$ contains the element '*'. But, since $\omega_*(\underline{r}) = \omega(\underline{r})$, this is a contradiction. Hence (13) holds, and $\underline{t}$ is definite. This proves T15.

For the remaining theorems we need the notions of contradictory formula and tautology.†

D13a. $\underline{s}$ is contradictory if and only if for every data base $\omega(\underline{s})$ is 0 (if $\underline{s}$ is of degree zero) or null (if $\underline{s}$ is of degree greater than zero).

D13b. $\underline{s}$ is a tautology if and only if $\underline{s}$ is of degree zero and $\omega(\underline{s})$ is 1 for every data base.

Examples: The formulas '(B,x)BN(B,x)' and 'NØT(T)' are both contradictory. The formulas '(B,b)ØR(NØT(B,b))' and 'T' are both tautologies.

---

†We use terminology consistent with Ref. 7. The two notions are not antithetic. The antithesis of contradictory is analytic; the antithesis of tautology is contradiction.

T16. If $\underline{r}$ and $\underline{s}$ are definite and neither is contradictory,† then:

$(\underline{r})\emptyset R(\underline{s})$ is definite if and only if $\bar{\varphi}(\underline{r}) = \bar{\varphi}(\underline{s})$.

Proof. Let $\underline{t}$ be $(\underline{r})\emptyset R(\underline{s})$. If $\underline{t}$ is of degree zero, then the theorem is true by T10. Therefore, let $\underline{t}$ be of degree $m > 0$.

A. For the implication from right to left, let $\bar{\varphi}(\underline{r}) = \bar{\varphi}(\underline{s})$. By T14 we must show that $\omega_*(\underline{t})$ cannot have a $\underline{b}^m$ containing '*' (as in the proof of T15). By D8

$$\omega_*([\varphi(\underline{t}) \rightarrow \underline{b}^m](\underline{t})) = 1$$

means that

$$\omega_*(\underline{r}_i) = 1 \text{ or } \omega_*(\underline{s}_j) = 1 \qquad (14)$$

Now, since $\bar{\varphi}(\underline{r}) = \bar{\varphi}(\underline{s})$, the induced substitutions $\underline{b}_i^p$ and $\underline{b}_j^q$ differ from each other and $\underline{b}^m$ by only a rearrangement. But then either of the statements in (14) implies that '*' cannot be in $\underline{b}^m$. Consider the first. If $\omega_*(\underline{r}_i) = 1$, then $\underline{b}_i^p \in \omega_*(\underline{r})$. Because $\underline{r}$ is definite, '*' is therefore not in $\underline{b}_i^p$ and hence not in $\underline{b}^m$.

---

†T16, T17, T18, T19, and T22 all have weak additional hypotheses (i.e., beyond definitude of the components). In each case, the additional hypothesis is required only to prove that the characterization in terms of free variables is a necessary condition of definitude.

B. To prove that t is definite implies $\bar{\varphi}(\underline{r}) = \bar{\varphi}(\underline{s})$, we show the contraposit. Suppose $\bar{\varphi}(\underline{r}) \neq \bar{\varphi}(\underline{s})$, then one component, say $\underline{r}$, has a $\underline{v}$ not in $\bar{\varphi}(\underline{s})$. By hypothesis and D13a, we can choose a data base $\mathfrak{B}_1$ in which $\omega(\underline{s})$ is either 1 (if $\underline{s}$ is of degree zero) or not null. Next construct the following substitution schema: If $\underline{s}$ is of degree zero, take

$$[\varphi(\underline{t}) \rightarrow *, *, \ldots, *]$$

If $\underline{s}$ is of degree $q > 0$, let $\underline{b}_j^q \in \omega(\underline{s})$ and then take

$$[\varphi(\underline{t}) \rightarrow \underline{b}^m] \tag{15}$$

where the simple substitutions are such that (15) entails

$$[\varphi(\underline{s}) \rightarrow \underline{b}_j^q]$$

and the substitution of '*' for each of the remaining variables (in particular, we have $[\underline{v} \rightarrow *]$). Since $\underline{s}$ is definite, we have for the data base $\mathfrak{B}_1$

$$\omega(\underline{s}_j) = \omega_*(\underline{s}_j) = 1$$

It now follows from D8 that

$$\omega_*([\varphi(\underline{t}) \rightarrow \underline{b}^m](\underline{t})) = 1$$

Hence, by D10, $\underline{b}^m \in \omega_*(\underline{t})$ with $\underline{b}^m$ containing '*'. Thus $\underline{t}$ is not semi-definite on $\mathfrak{B}_1$ and is therefore indefinite.

This proves T16.

T17. If $\underline{r}$ and $\underline{s}$ are definite and $\underline{r}$ is non-contradictory, then:

$(\underline{r})BN(\underline{s})$ is definite if and only if $\bar{\varphi}(\underline{s}) \subset \bar{\varphi}(\underline{r})$.

Proof. The proof is similar to that of T16. Let $\underline{t}$ be $(\underline{r})BN(\underline{s})$. To show the implication from the right to left, we use the same argument as in T16. However, in place of (14), we get

$$\omega_*(\underline{r}_i) = 1 \text{ and } \omega_*(\underline{s}_j) = 0 \qquad (16)$$

Now, $\bar{\varphi}(\underline{s}) \subset \bar{\varphi}(\underline{r})$ implies that $\bar{\varphi}(\underline{r}) = \bar{\varphi}(\underline{t})$. Hence, by (16) and the definitude of $\underline{r}$, a $\underline{b}^m$ with '*' cannot belong to $\omega_*(\underline{t})$. To show the implication in the other direction, we again show the contraposit. That is, if $\bar{\varphi}(\underline{s})$ is not contained in $\bar{\varphi}(\underline{r})$, then we can select $\underline{v}$ in $\bar{\varphi}(\underline{s})$ but not in $\bar{\varphi}(\underline{r})$. Next, we choose a data base for which $\omega(\underline{r})$ is either 1 (if $\underline{r}$ is of degree zero) or not null. Finally, we construct a substitution schema in which $\underline{v}$ is replaced by '*' and the induced substitution in $\underline{r}$ gives $\omega(\underline{r}_i) = 1$. As before, this forces $\omega_*(\underline{t})$ to contain a $\underline{b}^m$ with '*'. Hence $\underline{t}$ will not be semi-definite on this data base. This proves T17.

T18. If $\underline{r}$ and $\underline{s}$ are definite and $\underline{r}$ is not a tautology, then

$(\underline{r})IMP(\underline{s})$ is definite if and only if $\bar{\varphi}(\underline{r}) = \bar{\varphi}(\underline{s})$ = the null set.

Proof. Let $\underline{t}$ be $(\underline{r})\mathrm{IMP}(\underline{s})$. The implication from right to left follows immediately from T10. Suppose then, that $\underline{t}$ is definite. Then $\bar{\varphi}(\underline{r})$ must be null. For if it were not, then we could set up a substitution schema $[\varphi(\underline{t}) \rightarrow \underline{b}^m]$ in which a variable of $\bar{\varphi}(\underline{r})$ is replaced by '*'. We would then have, since $\underline{r}$ is definite, $\underline{b}_i^p \notin \omega_*(\underline{r})$, and hence that $\omega_*(\underline{r}_i) = 0$. By D8, this would lead to

$$\omega_*([\varphi(\underline{t}) \rightarrow \underline{b}^m](\underline{t})) = 1$$

and by D10 to $\underline{b}^m \in \omega_*(\underline{t})$, which contradicts the definitude of $\underline{t}$. To show that $\bar{\varphi}(\underline{s})$ is also null, choose a data base $\mathfrak{B}_1$ in which $\omega(\underline{r}) = 0$ (D13b). Hence $\omega_*(\underline{r}) = 0$. Then if $\bar{\varphi}(\underline{s})$ contains a variable $\underline{v}$, set up a substitution schema entailing the replacement of $\underline{v}$ by '*'. Then by D8, $\underline{b}^m \in \omega_*(\underline{t})$ and so $\underline{t}$ is not semi-definite on $\mathfrak{B}_1$, again contradicting the definitude of $\underline{t}$. This proves T18.

T19. If $\underline{r}$ and $\underline{s}$ are definite and neither is a tautology, then

$(\underline{r})\mathrm{IFF}(\underline{s})$ is definite if and only if $\bar{\varphi}(\underline{r}) = \bar{\varphi}(\underline{s}) =$ the null set.

Proof. Let $\underline{t}$ be $(\underline{r})\mathrm{IFF}(\underline{s})$. The implication from right to left follows by T10. Suppose then, that $\underline{t}$ is definite. Now if both $\bar{\varphi}(\underline{r})$ and $\bar{\varphi}(\underline{s})$ contain variables, then we can choose a substitution schema which replaces these variables by '*'. If $[\varphi(\underline{t}) \rightarrow \underline{b}^m]$ is this schema, then we would have for the induced substitution instances $\underline{r}_i$ and $\underline{s}_j$ the result

$$\omega_*(\underline{r}_i) = 0 \text{ and } \omega_*(\underline{s}_j) = 0$$

(since both $\underline{r}$ and $\underline{s}$ are definite). Hence, by D8,

$$\omega_*([\varphi(\underline{t}) \rightarrow \underline{b}^m](\underline{t})) = 1 \qquad (17)$$

Thus, $\underline{b}^m \in \omega_*(\underline{t})$, contradicting the definitude of $\underline{t}$. Consequently, if $\underline{t}$ is definite, then at least one component does not contain a free variable. Let $\bar{\varphi}(\underline{r})$ be null. By hypothesis and D13b, we can choose a data base $\mathfrak{B}_1$ for which $\omega(\underline{r}) = 0$. Since $\underline{r}$ is definite, we have

$$\omega_*(\underline{r}) = 0 \qquad (18)$$

Now if $\bar{\varphi}(\underline{s})$ contains a variable, then we can construct a substitution schema $[\varphi(\underline{t}) \rightarrow \underline{b}^m]$ that replaces this variable by '*'. Hence, since $\underline{s}$ is definite, we must have for the induced substitution instance $\underline{s}_j$

$$\omega_*(\underline{s}_j) = 0 \qquad (19)$$

By (18) and (19), the substitution $[\varphi(\underline{t}) \rightarrow \underline{b}^m]$ leads again to (17). Consequently, $\underline{t}$ is not semi-definite on $\mathfrak{B}_1$. This proves T19.

We now examine formulas of the form $\underline{\alpha}(\underline{s})$, where $\underline{\alpha}$ is one of the singulary operators with variables: $(E\underline{v})$, $(A\underline{v})$, $(P\underline{b}^n)$.

T20. If $\underline{s}$ is definite, then:

$(E\underline{v})(\underline{s})$ is definite.

Proof. Let $\underline{t}$ be $(E\underline{v})(\underline{s})$. If v does not occur in $\bar{\varphi}(\underline{s})$, then by D9 and D10 we have $\omega(\underline{t}) = \omega(\underline{s})$, and the theorem is true. Now let $\underline{v}$ occur in $\bar{\varphi}(\underline{s})$. If $\underline{s}$ is of degree one, then D9b gives

$$\omega_*(\underline{t}) = 0, \text{ if } \omega_*(\underline{s}) \text{ is null}$$

and

$$\omega_*(\underline{t}) = 1, \text{ if } \omega_*(\underline{s}) \text{ is not null.}$$

But, since $\omega_*(\underline{s}) = \omega(\underline{s})$, we then have

$$\omega_*(\underline{t}) = \omega(\underline{t}) \qquad (20)$$

Now let $\underline{s}$ be of degree $m > 1$. By T4, we have that $\underline{b}_o^{m-1} \in \omega_*(\underline{t})$ if and only if there is a certain $\underline{b}_o^m$ in $\omega_*(\underline{s})$. But since $\omega_*(\underline{s}) = \omega(\underline{s})$, it follows that $\underline{b}_o^{m-1} \in \omega(\underline{t})$. Hence $\omega_*(\underline{t})$ is included in $\omega(\underline{t})$. Similarly, the inclusion in the other direction is established. Consequently, (20) again holds, and $\underline{t}$ is definite. This proves T20.

T21. If $\underline{s}$ is definite and $\underline{v} \in \bar{\omega}(\underline{s})$, then:

$(A\underline{v})(\underline{s})$ is definite if and only if $(A\underline{v})(\underline{s})$ is contradictory.

Proof. Let $\underline{t}$ be $(A\underline{v})(\underline{s})$. A. For the implication from left to right, suppose that $\underline{t}$ is definite. First, consider $\underline{t}$ to be of degree zero. Then $\underline{s}$ is of degree one. Now, by hypothesis, $\underline{s}$ is definite. Thus '*' is not in $\omega_*(\underline{s})$, and D9b gives, for the calculation of $\omega_*(\underline{t})$,

$$\omega_*(t) = 0$$

But $\underline{t}$ is definite, therefore

$$\omega(\underline{t}) = 0$$

Next, let $\underline{t}$ be definite of degree $m > 0$, and suppose there were a $\underline{b}^m$ in $\omega(\underline{t})$. Then, by D10, we would have

$$\omega([\varphi(\underline{t}) \rightarrow \underline{b}^m](\underline{t})) = 1 \qquad (21)$$

But, by T8, $[\varphi(\underline{t}) \rightarrow \underline{b}^m](\underline{t})$ is a definite formula of degree zero; and, by D6, it is of the form $(A\underline{v})(\underline{s}_i)$, where (again by T8) $\underline{s}_i$ is definite and of degree one. Hence (21) contradicts the result obtained for the previous case. So $\omega(\underline{t})$ is null. Consequently, by D13a, $\underline{t}$ is contradictory.

B. For the implication from right to left, let $\underline{t}$ be contradictory. Consider first the case where $\underline{t}$ is of degree zero. Then, since $\omega(\underline{t}) = 0$, it follows from D9b that there is a $\underline{d}$ not in $\omega(\underline{s})$. Hence, there is a $\underline{d}$ not in $\omega_*(\underline{s})$. Consequently, we have for $\omega_*(\underline{t})$ (by D9b)

$$\omega_*(\underline{t}) = 0$$

Thus $\underline{t}$ is definite. For the other case, let $\underline{t}$ be contradictory and of degree $m > 0$. Hence $\omega(\underline{t})$ is null. Now suppose there were a $\underline{b}^m$ in $\omega_*(\underline{t})$. Then, by D9b, every $\underline{d}$ together with '*' would be in $\omega_*(\underline{s}_i)$ where $\underline{s}_i$ is $[\varphi(\underline{t}) \rightarrow \underline{b}^m](\underline{s})$. This means that there is some $\underline{b}^{m+1}$ with '*' belonging to $\omega_*(\underline{s})$. But this contradicts the

definitude of $\underline{s}$. Hence $\omega_*(\underline{t})$ is null, as is $\omega(\underline{t})$, and so $\underline{t}$ is definite.

This proves T21.[†]

Note that T21 tells us that nothing useful comes from a non-vacuous universal quantification of a definite formula. Useful definite formulas of the form $(A\underline{v})(\underline{s})$ must therefore stem from a component $\underline{s}$ that is <u>indefinite</u>. This is indeed the case. The most important use of $(A\underline{v})$ is with an implication[††] as component, and, by T18, implications are indefinite when they contain free variables. (In Sec. 5 we will examine the definitude of universal implications.)

<u>T22</u>. If $\underline{s}$ is definite and non-contradictory, then:

$(P\underline{b}^n)(\underline{s})$ is definite if and only if $\bar{\varphi}(\underline{b}^n) \subset \bar{\varphi}(\underline{s})$.

Proof. Let $\underline{t}$ be $(P\underline{b}^n)(\underline{s})$. A. The implication from right to left follows immediately. This is because $\underline{t}$ cannot be of degree zero and, by D10 and D5b, the members of $\omega(\underline{t})$ are permuted members of $\omega(\underline{s})$.

B. Now let $\underline{t}$ be definite and suppose there is a variable $\underline{v}$ in $\underline{b}^n$ that is not in $\bar{\varphi}(\underline{s})$. Since $\underline{s}$ is non-contradictory, we can select a data base $\mathfrak{D}_1$ in which

---

[†] In part A we actually proved that $(A\underline{v})(\underline{s})$ is contradictory in a stronger sense (i.e., including the pseudo data bases). In part B, on the other hand, we assumed 'contradictory' in the weaker sense of D13a.

[††] The use with equivalence is secondary because equivalence is co-valued with a conjunction of implications and universal quantification is distributive over conjunction.

$\omega(\underline{s}) = 1$ (if $\underline{s}$ is of degree zero) or in which $\omega(\underline{s})$ is not null (if $\underline{s}$ is of degree greater than zero). In either case, we can construct a substitution schema $[\varphi(\underline{t}) \rightarrow \underline{b}^m]$ (as in part B of the proof of T16) which replaces $\underline{v}$ by '*' and yields

$$\omega_*(\underline{s}_i) = \omega(\underline{s}_i) = 1$$

Hence $\omega_*(\underline{t})$ contains a member in which '*' occurs. This contradicts the definitude of $\underline{t}$.

This proves T22.

4.5 THE CHARACTERIZATION THEOREM FOR PROPER FORMULAS

In the previous section we studied definite formulas with definite components. The results are given by T13(NØT), T15(AND), T16(ØR), T17(BN), T18(IMP), T19(IFF), T20((E$\underline{v}$)), T21((A$\underline{v}$)), T22((P$\underline{b}$)). Each of these theorems has the following general form. If the components of a formula are definite (and possibly other conditions are fulfilled), then: the formula is definite if and only if a certain characterization holds. The additional hypotheses occur in T16, T17, T18, T19, and T22[†] and are required only to prove the necessity of the characterization. Now, by definition (D12), non-elementary proper formulas are definite with definite components. It therefore follows that each of the theorems has as a corollary the corresponding theorem with 'definite' replaced by 'proper'.

[†] T21 is not listed because the additional hypothesis merely states that the quantification is not vacuous.

Hence we can derive the following characterization theorem for proper formulas. This theorem gives a set of machine-recognizable sufficient conditions for propriety.

T23. If r and s are proper, then:

a. (Ev)(r) and (r)AND(s) are proper.

b. If $\bar{\varphi}(s) \subset \bar{\varphi}(r)$, then (r)BN(s) is proper.

c. If $\bar{\varphi}(r) = \bar{\varphi}(s)$, then (r)ØR(s) is proper.

d. If $\bar{\varphi}(r) = \bar{\varphi}(s)$ = the null set, then NØT(r), (r)IMP(s), and (r)IFF(s) are proper.

e. If $\bar{\varphi}(b) \subset \bar{\varphi}(r)$, then (Pb)(r) is proper.

Note that the results a, b, c, d are successively weaker; i.e., the hypotheses are successively stronger. Thus, the hypothesis of d implies the hypothesis of c which, in turn, implies the hypothesis of b. Finally, a, the strongest result, holds under any condition.

The characterization theorem also gives the weakest set of sufficient conditions. This is because of the following companion theorem on necessary conditions.

T24. The conditions listed in the characterization theorem are necessary for propriety with the following additional provisions:

a. For (r)BN(s) and (Pb)(r), providing r is not contradictory.

b. For (r)ØR(s), providing neither r nor s is contradictory.

c. For (r)IMP(s), providing r is not a tautology.

d. For (r)IFF(s), providing neither r nor s is a tautology.

4.6 VALUE SETS OF PROPER FORMULAS

Given a certain formula, suppose that we have applied the characterization theorem (T23) and determined that it is proper. We now want to calculate its value set. However, from a computational standpoint, not all of our previous definitions are suitable for this purpose. Therefore, we now develop the required rules of computation. These are stated in terms of program-like algorithms for computer implementation.

Array Representations. The inputs to the computation of $\omega(\underline{s})$, where $\underline{s}$ is not elementary, are the value sets of the components. The computer representation of these inputs are taken to be arrays as discussed in Sec. 3.5. That is, a formula $\underline{s}$ of degree zero has its value set represented by a 2×1 array A with

$$A(1,1) = \Delta,\ A(2,1) = \omega(\underline{s}) \tag{22}$$

A formula $\underline{s}$ of degree $m > 0$ has its value set represented by an n×m array A with

$$A(1,j) = j^{th} \text{ element in } \omega(\underline{s}) \tag{23}$$

and, for $i > 1$,

$$A(i,j) = j^{th} \text{ element in } \underline{b}^m, \text{ where } \underline{b}^m \text{ is the } (i-1)^{th} \text{ member of } \omega(\underline{s}) \text{ under a certain ordering.} \tag{24}$$

The first row of A we call the heading row, and the others, data rows.

The output of a computation has a similar representation.

Procedure for Elementary Formulas. For the case of an elementary formula, the input is simply the formula itself. This too can be regarded as an array, i.e., a 1xn array, where n is the length of the formula. The value set is calculated by a direct match with the file. This is done in the following steps.

First, we represent the input formula $\underline{f}$ as an array. Second, an output array W is set up. Thus the first row of W will be $\varphi(\underline{f})$. We now extract from the file those elementary formulas whose elements coincide with the descriptive names of $\underline{f}$ at each position of $\underline{f}$ in which a descriptive name occurs. If $\underline{f}$ is of degree zero and a match is found, then W(2,1) = 1; otherwise, W(2,1) = 0. If $\underline{f}$ is of degree m > 0, then, for each extracted elementary sentence, we evaluate the variables of $\varphi(\underline{f})$ by equating them with the elements that occur in the corresponding positions of the extracted sentence. If multiple occurrences of variables give consistent values, then the evaluation is entered into a row of W as a member of $\omega(\underline{f})$.

Procedure for Singulary Formulas. Turning now to the non-elementary formulas, we consider first the singulary formulas; i.e., those of the design $\underline{\alpha}(\underline{s})$.

Let the input array A represent $\omega(\underline{s})$, and let the output array W represent $\omega(\underline{\alpha}(\underline{s}))$. By T23, three cases are possible: $\underline{\alpha}$ is $(E\underline{v})$; $\underline{\alpha}$ is $(P\underline{b})$, with $\bar{\varphi}(\underline{b}) \subset \bar{\varphi}(\underline{s})$; $\underline{\alpha}$ is NØT, with $\bar{\varphi}(\underline{s})$ being the null set. The rules corresponding to these situations are:

Rule 1. (E$\underline{v}$)($\underline{s}$).

Let the input array A have dimension n×m. We have three cases:

$\underline{a}$. If $A(1,j) \neq \underline{v}$ for any j, then W = A.

$\underline{b}$. If m = 1 and $A(1,1) = \underline{v}$, then $W(1,1) = \Delta$; and if n = 1,

$$W(2,1) = 0$$

otherwise (n > 1),

$$W(2,1) = 1$$

$\underline{c}$. If m > 1 and $A(1,j) = \underline{v}$ for some j, then W is obtained from A by first deleting the j^{th} column of A, and then eliminating repetitions of rows in the result.

Rule 2. (P$\underline{b}$)($\underline{s}$).

W is obtained from A by permuting the columns of A so that

$$W(1,j) = j^{th} \text{ element in } \varphi(\underline{b},\varphi(\underline{s}))$$

Rule 3. NØT($\underline{s}$).

We set

$$W(1,1) = \Delta,\ W(2,1) = 1 - A(2,1)$$

Rule 1 is justified by D9 and T4; Rule 2 by D5b, D10, D6b-4; and Rule 3 by D8a.

Standard Forms. We next consider binary formulas, i.e., those of the design $(\underline{r})\beta(\underline{s})$. Let the input arrays A and B represent $\omega(\underline{r})$ and $\omega(\underline{s})$, respectively. Let the output array W represent $\omega((\underline{r})\beta(\underline{s}))$.

Some auxiliary calculations are now introduced. These are the determinations of the variables in $\varphi(r)$ but not in $\varphi(\underline{s})$, the variables in both, and the variables in $\varphi(\underline{s})$ but not in $\varphi(\underline{r})$. The results, when listed as free variable sequences, are denoted by 'X', 'Y', and 'Z', respectively. They are useful for two purposes. The first is to identify the type of calculation required. The second is to bring the arrays A and B to a certain standard form. The idea of a standard form derives from the following considerations. Each row of A can be divided into two subsequences corresponding to the two subsequences X and Y of $\varphi(\underline{r})$. It turns out that in the calculation of a value set these subsequences act as single elements corresponding to the pseudo-variables 'X' and 'Y'. Thus A can be considered to be an n×2 array with A(1,1) = X, A(1,2) = Y, and whose other elements are subsequences. B can be regarded similarly. We now develop the technical details.

D14. Auxiliary Variable Sequences. Given $(\underline{r})\beta(\underline{s})$, let

$$X = \delta(\varphi(\underline{r});\ \varphi(\underline{s}))$$

$$Y = \delta(\varphi(\underline{r});\ X)$$

$$Z = \delta(\varphi(\underline{s});\ Y)$$

D15. Standard Forms. Given $(\underline{r})\underline{\beta}(\underline{s})$, let

$$\underline{r}_o = (PX,Y)(\underline{r})$$

if $\bar{\varphi}(\underline{r})$ is not null; otherwise let $\underline{r}_o = \underline{r}$. Similarly, let

$$\underline{s}_o = (PY,Z)(\underline{s})$$

if $\bar{\varphi}(\underline{s})$ is not null; otherwise let $\underline{s}_o = \underline{s}$. The standard forms for A and B are then the array representations of $\omega(\underline{r}_o)$ and $\omega(\underline{s}_o)$, respectively.

The standard forms can be used in the calculation of $\omega((\underline{r})\underline{\beta}(\underline{s}))$ without loss of generality. This is because

$$\omega((\underline{r})\underline{\beta}(\underline{s})) = \omega((P\varphi(\underline{r}),\varphi(\underline{s}))((\underline{r}_o)\underline{\beta}(\underline{s}_o))) \qquad (25)$$

Classification of Binary Formulas. We next use X, Y, and Z to classify the formulas of design $(\underline{r})\underline{\beta}(\underline{s})$. The classification corresponds to parts a, b, c, and d of the characterization theorem (T23).

If $Z \neq \Delta$, then the formula is of Class A, and $\underline{\beta}$ must be AND.

If $Z = \Delta$, $X \neq \Delta$, then the formula is of Class B, and $\underline{\beta}$ must be either AND or BN.

If $Z = \Delta$, $X = \Delta$, $Y \neq \Delta$, then the formula is of Class C, and $\underline{\beta}$ must be either AND, BN, or ØR.

If $Z = \Delta$, $X = \Delta$, $Y = \Delta$, then the formula is of Class D, and all five $\underline{\beta}$'s are possible.

An inventory of binary formulas and index of the rules of computation is given in Table 3. A dash in the Z, X, or Y column means the value is not Δ. In the column labelled 'Model', we give analogous formulas in the predicate calculus whose components are of degree zero, one, or two. Three types of expressions are used to represent these components. A component of degree one or two is represented by a predicate followed by either one or two variables, respectively. For example, 'Fx', a one-place predicate expression, represents a component of degree one; 'Fxy', a two-place predicate expression, represents a component of degree two. Since a component of degree zero (i.e., a sentence) has no inner structure of interest to us, it is represented by 'p' or 'q'. To simplify the notation, the usual parentheses used to enclose components will be omitted. The two-place predicate expressions correspond to an array of two columns. Borrowing terminology from the logic of relations [8, §§30-38], we call the distinct elements in the first column (other than the variable) the <u>domain</u>, and the distinct elements in the second column (other than the variable) the <u>converse domain</u>. Similarly, the one-place predicate expressions correspond to a columnar array headed by a variable. Finally, the sentences correspond to a 1x2 array headed by Δ. Thus the models indicate the type of array under consideration. They are also given to illustrate the type of operation required for the computation rules. In the last column we give a generic name to the operation involved. These operations and the illustrative models will be discussed in more detail in Sec. 4.7 below, but the

Table 3

CLASSIFICATION OF FORMULAS OF THE DESIGN $(\underline{r})\underline{\beta}(\underline{s})$ †

Z	X	Y	Class	$\underline{\beta}$	Rule	Model	Operation
-	-	-	A	AND	4	Fxy AND Gyz	Cartesian
-	-	Δ	A	AND	5	Fx AND Gz	Cartesian
-	Δ	-	A	AND	6	Fy AND Gyz	Restrictive
-	Δ	Δ	A	AND	7	p AND Gz	Restrictive
Δ	-	-	B	AND	8	Fxy AND Gy	Restrictive
Δ	-	-	B	BN	9	Fxy BN Gy	Restrictive
Δ	-	Δ	B	AND	10	Fx AND q	Restrictive
Δ	-	Δ	B	BN	11	Fx BN q	Restrictive
Δ	Δ	-	C	AND	12	Fy AND Gy	Class
Δ	Δ	-	C	BN	13	Fy BN Gy	Class
Δ	Δ	-	C	ØR	14	Fy ØR Gy	Class
Δ	Δ	Δ	D	AND	15	p AND q	Arithmetic
Δ	Δ	Δ	D	BN	16	p BN q	Arithmetic
Δ	Δ	Δ	D	ØR	17	p ØR q	Arithmetic
Δ	Δ	Δ	D	IMP	18	p IMP q	Arithmetic
Δ	Δ	Δ	D	IFF	19	p IFF q	Arithmetic

†Dash in variable column means the value is not Δ.

classification is as follows. If neither X nor Z is Δ, then the operation is cartesian. If either X or Z (but not both) is Δ, then the operation is one of restriction. Thus Class A has both cartesian and restrictive operations and Class B has only the latter. The operations of Class C (both X and Z are Δ, but $Y \neq \Delta$) are the familiar class operations of intersection, difference, and union. Finally, in Class D (X, Y, and Z are Δ) we have the Boolean arithmetic operations.

Procedure for Binary Formulas. The rules of computation for the binary formulas are now presented. We follow the classification of Table 3. A discussion of the significance of these rules in regard to the models is given in Sec. 4.7 below.

Rule 4. (r)AND(s), with $X \neq \Delta$, $Y \neq \Delta$, $Z \neq \Delta$.

1) Determine the values of Y common to both A and B; i.e., obtain the intersection† S of the columnar arrays A(-,2) and B(-,1). Thus S is a set of distinct values and can be represented as a columnar array.

2) Delete from A all rows A(i,-) for which $A(i,2) \notin S$ and delete from B all rows B(i,-) for which $B(i,1) \notin S$. Let A' and B' be the resulting arrays.

3) Suppose now that k rows of W have been filled and i-1 rows of A' have been processed. To process the i^{th} row of A', define $\ell_1, \ldots, \ell_{k_i}$ to be the row indices of B' for which

$$B'(\ell_j,1) = A'(i,2)$$

†Y itself will be in this intersection.

Then, the next k_i rows of W are filled according to the formulas: for $j = 1, \ldots, k_i$

$$W(k+j,1) = A'(i,1)$$

$$W(k+j,2) = A'(i,2)$$

$$W(k+j,3) = B'(\ell_j,2)$$

A crude estimate of the size of the output array given by Rule 4 is $n_1 n_2/n$ data rows where n_1, n_2 are the number of data rows in A' and B', respectively, and n is the number of data elements in S.

Because Rule 4 yields cartesian-like products of sections of the domain of A with sections of the converse domain of B, we call the result the <u>partial cartesian product</u> and class the operation as <u>cartesian</u>.

The next situation gives the traditional cartesian product.

<u>Rule 5</u>. (<u>r</u>)AND(<u>s</u>), with $X \neq \Delta$, $Y = \Delta$, $Z \neq \Delta$.

Initially, we set W(1,1) = X, W(1,2) = Z. Let A and B have n_1, n_2 data rows, respectively. We next suppose that k rows of W have been filled and i-1 data rows of A have been processed. Then, the i^{th} data row of A yields n_2 rows of W according to the formulas: for $j = 1, \ldots, n_2$

$$W(k+j,1) = A(i+1,1)$$

$$W(k+j,2) = B(j+1,1)$$

This computation gives an output array with $n_1 n_2$ data rows.

Rule 6. (r)AND(s), with $X = \Delta$, $Y \neq \Delta$, $Z \neq \Delta$.

1) Form S as in Rule 4 (i.e., intersect the columnar arrays A and B(-,1)).
2) Form B' as in Rule 4 (i.e., delete from B all rows B(i,-) for which B(i,1) $\notin$ S).
3) Then

$$W = B'$$

The result of Rule 6 is therefore a subarray of B obtained by restriction of the domain.

Rule 7. (r)AND(s), with $X = \Delta$, $Y = \Delta$, $Z \neq \Delta$.

If A(2,1) = 1, then W = B; otherwise W is the 1x1 array with W(1,1) = Z.

Rule 7 gives the restricted occurrences of the value set of s. That is, (r)AND(s) is co-valued with s providing $\omega(r) = 1$, otherwise its value set is null.

Rule 8.† (r)AND(s), with $X \neq \Delta$, $Y \neq \Delta$, $Z = \Delta$.

1) Form S as in Rule 4 (i.e., intersect the columnar arrays A(-,2) and B).
2) Form A' as in Rule 4 (i.e., delete from A all rows A(i,-) for which A(i,2) $\notin$ S).
3) Then

$$W = A'$$

---

†An alternate rule is: interchange r and s and use Rule 6. Presumably, however, the "bookkeeping" on this would be more difficult than the execution of Rule 8.

This is similar to Rule 6, except that a subarray of A results by restriction of the converse domain.

Rule 9. (r)BN(s), with $X \neq \Delta$, $Y \neq \Delta$, $Z = \Delta$.

1) Form S as in Rule 4 (i.e., intersect the columnar arrays A(-,2) and B).
2) Form A' by deleting from A all data rows A(i,-) for which $A(i,2) \in S$.
3) Then

$$W = A'$$

Again, we have a subarray of A as the result, but the criterion for selection in this case is a negative restriction of the converse domain.

The next two rules, like Rule 7, restrict the occurrence of a value set.

Rule 10. (r)AND(s), with $X \neq \Delta$, $Y = \Delta$, $Z = \Delta$.

If B(2,1) = 1, then W = A; otherwise W is the 1x1 array with W(1,1) = X.

Rule 11. (r)BN(s), with $X \neq \Delta$, $Y = \Delta$, $Z = \Delta$.

If B(2,1) = 0, then W = A; otherwise W is the 1x1 array with W(1,1) = X.

The next three rules give the class operations of intersection, difference, and union.

Rule 12. (r)AND(s), with $X = \Delta$, $Y \neq \Delta$, $Z = \Delta$.

Form S as in Rule 4 (i.e., intersect the columnar arrays A and B), then

$$W(i,1) = S(i)$$

Rule 13. (r)BN(s), with $X = \Delta$, $Y \neq \Delta$, $Z = \Delta$.

1) Form S as in Rule 4 (i.e., intersect the columnar arrays A and B).
2) Form A' by deleting from A all data rows A(i,1) for which $A(i,1) \in S$.
3) Then

$$W = A'$$

Rule 14. (r)ØR(s), with $X = \Delta$, $Y \neq \Delta$, $Z = \Delta$.

1) Annex the array B to A; i.e., form the array C according to the formulas: if $1 \leq i \leq n_1$,

$$C(i,1) = A(i,1)$$

and if $n_1+1 \leq i \leq n_1+n_2$,

$$C(i,1) = B(i-n_1,1)$$

where n_1, n_2 are the number of rows in A and B, respectively.

2) W is now obtained from C by eliminating repetitions of rows.

Rules 4-14 are justified by D8 and D10.

Finally, we have the Boolean arithmetic operations. These are given directly by D8, but are repeated here. In these rules (15-19) we calculate first the Boolean symmetric difference,

$$q = |A(2,1) - B(2,1)|$$

Rule 15. (r)AND(s), with X = Δ, Y = Δ, Z = Δ.

$$W(1,1) = \Delta$$

$$W(2,1) = \tfrac{1}{2}[A(2,1) + B(2,1) - q]$$

Rule 16. (r)BN(s), with X = Δ, Y = Δ, Z = Δ.

$$W(1,1) = \Delta$$

$$W(2,1) = \tfrac{1}{2}[A(2,1) - B(2,1) + q]$$

Rule 17. (r)ØR(s), with X = Δ, Y = Δ, Z = Δ.

$$W(1,1) = \Delta$$

$$W(2,1) = \tfrac{1}{2}[A(2,1) + B(2,1) + q]$$

Rules 15, 16, and 17 yield the Boolean product, difference, and sum, respectively.

Rule 18. (r)IMP(s), with X = Δ, Y = Δ, Z = Δ.

$$W(1,1) = \Delta$$

$$W(2,1) = 1 - \tfrac{1}{2}[A(2,1) - B(2,1) + q]$$

Rule 19. (r)IFF(s), with X = Δ, Y = Δ, Z = Δ.

$$W(1,1) = \Delta$$

$$W(2,1) = 1 - q$$

Rules 18 and 19 are complementations; they yield the complements of the Boolean arithmetic difference and symmetric difference, respectively.

4.7 PROPER FORMULAS, LANGUAGE, AND THE LOGIC OF RELATIONS

We have developed a theory of proper formulas and established procedures for its implementation by means of machine-executable rules for the calculation of value sets. It now remains to cast this theory in the frame of traditional logic. Of course, our main interest here is that part of logic which provides tools for the machine processing of natural language.

In the previous section, the first steps were taken by presenting examples of binary proper formulas in the predicate calculus. We continue by showing how certain important concepts in the logic of relations [8, §§30-38] are formulated in terms of proper formulas. This will also lead to a discussion of the meaning of value sets as expressions in an object language.†

Representations in the Predicate Calculus. In Sec. 4.6 above, expressions such as 'p', 'Fx', 'Fxy' were used to represent formulas of degree zero, one, and two, respectively. This is now explained in more detail.

Any formula of degree zero (i.e., a sentence) is indicated by the letters 'p' or 'q'. The meaning of the value set (or simply, the value) of such a formula is the truth value of the formula when considered as a sentence in the object language--'1' meaning true and '0' meaning false.

---

†See p. 7, footnote.

A formula of degree one is represented by an expression such as 'Fx', which is interpreted in the following way. First, we regard the original formula, say s with free variable 'x', as defining a certain property. For example, let s be the formula 'x is red and x is a book'. The property defined by s is that of being a red book; in other words, the property that x has when x is red and x is a book. In order to always have a convenient name for such properties, logic uses a special operator, the λ-operator [7]. We write

$$(\lambda x)(s) \qquad (26)$$

which is read as: the property that x has when s (is the case). The expression (26) is then abbreviated by 'F'. 'F', or its λ-expression, is therefore a one-place predicate designating a property. Of course, if the sentential formula is elementary with variable occurring in the second place (e.g., 'Bx' as in Sec. 1), then the predicate is simply the initial element and we do not need the λ-expression. The value set of s, as an expression (or a list of expressions) in the object language, inventories a class, the extension of the predicate 'F'.

Similarly, a sentential formula s of degree two is represented by an expression 'Fxy'. 'F' is a two-place predicate and designates a relation; namely, the relation that x has to y when s (is the case). In terms of the λ-operator we have the definition

$$F = (\lambda xy)(s)$$

The value set of the formula s then inventories a class of ordered pairs, the extension of the predicate 'F'.

We shall be primarily concerned with classes (properties) and relations which stem from relations.

Proper Reduction of Degree. First we have classes which arise when we reduce the degree of a formula.

By T23a, the use of an existential quantifier with a proper formula always leads to a proper formula. Two simple situations are of special interest here. Let 'Fxy' be proper (i.e., represent a proper formula). Then we have the two existential quantifications which yield formulas of degree one:

$$(Ey)(Fxy) \tag{27}$$

$$(Ex)(Fxy) \tag{28}$$

As expressions in the object language, the value sets of these formulas give the domain and converse domain, respectively, of the relation F.

For example, if F is the relation wrote (in the sense used in the example file of Table 1, p. 4), then the domain is authors and the converse domain is publications.

There is a second method of reducing the degree which is also always proper and leads to important classes. This is by substitution of a descriptive name (T9).† From 'Fxy' we get

†Recall that a separate rule was not given for calculating the value set of a substitution instance of a formula. It was not necessary to do so because substitution does not enter explicitly into the design of a formula.

$$Fxy_o \tag{29}$$

by substituting for 'y', and

$$Fx_oy \tag{30}$$

by substituting for 'x'. The value sets of (29) and (30) give the <u>referents</u> of y_o and the <u>relata</u> of x_o (in the relation F), respectively.

For example, again letting F be the relation <u>wrote</u>, the referents of y_o are the authors of y_o and the relata of x_o are the publications of x_o.

<u>Permutation and Converses</u>. Forming the <u>converse</u> of a relation is one of the two operations in the calculus of relations that have no analogy in the calculus of classes. The other is formation of a relative product, which will be discussed later. Both are important for the analysis of natural language.

The converse operation is denoted by '$\breve{}$'; its formal definition is

$$\breve{F} = (\lambda yx)(Fxy) \tag{31}$$

Thus $\breve{F}$ is the relation that y has to x when Fxy (is the case). It follows that

$$(Pyx)(Fxy) \tag{32}$$

is of special interest, since its value set describes the extension of '$\breve{F}$'. (An example of a converse relation is given below.)

Restricted Domains and Converse Domains. Turning to the binary formulas, we note that the situations discussed in Rules 6 and 8 (see Table 3, p. 73) are of particular significance. These have to do with a relation-forming operation between a class (property) and a relation. Letting F be the class (property) and G the relation, the defining formulas are:

$$_{F}{}^{G} = (\lambda xy)(Fx \text{ AND } Gxy) \qquad (33)$$

$$G_{F} = (\lambda xy)(Gxy \text{ AND } Fy) \qquad (34)$$

Formula (33) defines a relation with restricted domain; (34), a relation with restricted converse domain. For example, the relation father of is obtained by restricting the domain of the relation parent of to males.

The concepts of converse, restricted domain, and referents frequently arise in natural-language questions. For example, a question which involves all three is:

How many books has R. Carnap written? (35)

This means to count the referents of R. Carnap in the relation has been written by (the converse of has written) with domain restricted to books.

Proper Reduction of Degree (continued). Reducing the degree of the formula 'Fxy' led to the classes of domain, converse domain, referents, and relata. Let us now study the effect of reducing the degree of a binary formula.

Referring to the inventory given in Table 3 (p. 73), we first note that substitution for one of the variables always yields another proper binary formula. For example, substitution of a descriptive name for 'x' in

Fxy AND Gyz

(a model for Rule 4) gives a formula that can be modelled by

Fy AND Gyz

(a model for Rule 6), etc. Similarly, existential quantification of a variable occurring in only one component leads to another formula on the list.

Consider then the formulas with a common variable in each component. Only the disjunction 'Fy ØR Gy' is without interest (because existential quantification is distributive over disjunction). Among the others, the first is

Fxy AND Gyz

From this we get the relative product F|G of the relations F and G:

$$F|G = (\lambda xz)((Ey)(Fxy \text{ AND } Gyz)) \qquad (36)$$

The notion of relative product provides an important tool for semantic analysis. Consider, for example, the question:

Did Russell co-author with Whitehead? (37)

The example file (Table 1, p. 4) contains all the information necessary to answer this, but the information is not directly accessible. Therefore, we must mechanically connect the meaning of the two-place predicate co-author with occurring in (37) with the predicates of the file (in particular, the predicate 'W'). This is done by defining[†]

$$\text{co-author with} = W|\breve{W}$$

Next we have the formulas

Gy AND Fyz (38)

Fxy AND Gy (39)

Fxy BN Gy (40)

The existential quantification of (39) is treated first, since this is assigned a special notation in logic [8, §37]. The result is denoted by 'F"G'. We call this a plural relational description. Exactly,

$$F"G = (\lambda x)((Ey)(Fxy \text{ AND } Gy)) \quad (41)$$

Thus, F"G is the class of things having the relation F to things having the property G. For example, 'editors of

[†]More exactly, we should also specify the relation to be contained in diversity, i.e., no individual is co-author with himself.

mathematics journals' is a phrase which can be analyzed as a plural relational description.†

The existential quantifications of (38) and (40) also yield plural relational descriptions. From (38) we get '$\breve{F}$"G' involving the converse of F. From (40) we get 'F"$\bar{G}$' involving the negation, $\bar{G}$, of G (or, the complement, if G is regarded as a class). These expressions have less interest for us since they do not directly correspond to natural-language phrases.

Finally, we have the formulas

$$Fy \text{ AND } Gy \tag{42}$$

$$Fy \text{ BN } Gy \tag{43}$$

From (42) we get the formula

$$(Ey)(Fy \text{ AND } Gy) \tag{44}$$

which can properly be regarded as the <u>fundamental</u> use of the existential quantifier. The reason for this is the following. Language discusses certain "natural" properties (e.g., <u>blue</u>, <u>warm</u>, <u>books</u>, etc.). A simple quantification such as

---

†The notation 'F"G' used in Ref. 8 is indicative of the linguistic structure of plural relational descriptions. Thus the reversed double apostrophe can be read as 'of'. In natural language the presence of such a description is often flagged by 'of' in association with adjacent plurals.

Our term <u>plural relational description</u> is introduced in place of <u>plural descriptive function</u> [8]. This last term, in modern usage, properly refers to F" (i.e., the set function induced by the relation F).

(Ey)(Fy)

where F is one of these "natural" properties, is therefore not of much interest; it merely states that the property occurs. Existential quantification over disjunctions† are of even less interest. Consequently, only conjunctions yield something informative and this is why the formula (44) is intrinsically fundamental.

Of course, the existential quantifier does not always stem from 'some' or 'there is'; English usually masks it with the indefinite description [3]. This is an argument expression consisting of an indefinite article followed by a noun. Logic provides us with a special notation, the η-operator (together with a rule for its transformation), which results in symbolic translations closely matching natural-language structure. The transformation rule then yields a formula such as (44). For example, 'a book' is symbolixed as '$(\eta x)(Bx)$'. Thus, the sentence 'R. Carnap wrote a book' is symbolized as '$Wa(\eta x)(Bx)$' (see Table 1, p. 4). The transformation rule then gives the existential quantification of a conjunction, namely,

(Ex)(Wax AND Bx)

which has the form of (44).

---

†This includes implication and equivalence. For binary formulas concerning "natural" properties, the scale of decreasing interest would be existential quantification over disjunction, equivalence, and implication.

The last formula to be considered is (43). Because of the equivalence of BN with AND NØT, this can be classed with (42) as conjunctive. The existential quantification of (43) is

(Ey)(Fy BN Gy) (45)

This gains additional significance when we note that the negation of (45) is equivalent to

(Ay)(Fy IMP Gy) (46)

which can properly be regarded as the fundamental use of the universal quantifier. In the next section we will have more to say about (46). It belongs to the important class of formulas that are definite but improper.

5. DEFINITE FORMULAS AND ADMISSIBILITY

5.1 INITIAL REMARKS

In this section our goal is to establish procedures for transforming a definite but improper formula into a proper formula having the same value set. If this can be done in a way that is independent of the data base, then the formula is called admissible and its value set can be calculated by the methods shown in the previous section.

Unfortunately, we are only partially successful. It is still an open question whether all definite formulas are admissible. On the other hand, the limited results obtained are useful; we can prove the admissibility of definite formulas without quantifiers and the admissibility of certain formulas involving "natural" uses of universal quantifiers.

5.2 ADMISSIBILITY

The formal definition of admissible formula is:

D16. $\underline{r}$ is admissible if and only if there is a proper formula $\underline{s}$ such that for every data base

$$\omega(\underline{r}) = \omega(\underline{s}) \qquad (1)$$

To establish the admissibility of a formula it is convenient to make use of well-known logical equivalences. References to such equivalences have been made in the previous sections; now let us give some explanation.

In our terminology the logical equivalence of two formulas $\underline{r}$ and $\underline{s}$ has the following meaning: If m is the degree of $(\underline{r})\text{IFF}(\underline{s})$, then for every data base and every $\underline{b}^m$

$$\omega([\omega((\underline{r})\text{IFF}(\underline{s})) \rightarrow \underline{b}^m]((\underline{r})\text{IFF}(\underline{s}))) = 1 \qquad (2)$$

Suppose then that $\underline{r}$ and $\underline{s}$ are logically equivalent. If, in addition, the free variable sequences of $\underline{r}$ and $\underline{s}$ are the same, then, by D8b and D10, we can infer from (2) that $\underline{r}$ and $\underline{s}$ are <u>co-valued</u>.† That is, for every data base we have

$$\omega(\underline{r}) = \omega(\underline{s})$$

In this way, the criterion (1) is related to logical equivalence.

(The definition of admissibility is rather weak and should be regarded as tentative. In fact, under the narrow interpretation of "data base" (see Sec. 4.2 above), we need a special argument to show that an admissible formula is definite. This is because D16 gives the equations

$$\omega(\underline{r}) = \omega(\underline{s})$$

and

$$\omega(\underline{s}) = \omega_*(\underline{s})$$

---

†We cannot use the term 'equivalence' here. For example, the formulas 'W,x,y' and 'W,y,x' are co-valued, but certainly not equivalent (in the sense suggested by the term 'equivalent').

but not (at least not directly) the equation

$$\omega_*(\underline{r}) = \omega_*(\underline{s}) \qquad (3)$$

If (3) is indeed derivable from D16, then, presumably, a concept of isomorphic data bases is required in the derivation. This has subtle aspects not pertinent to our main purpose. In the particular cases with which we will be concerned, admissibility is always by way of a logical equivalence, and such a strong relation between $\underline{r}$ and $\underline{s}$ does give (3).)

5.3 DEFINITE FORMULAS WITHOUT QUANTIFIERS

We first solve the admissibility problem for formulas without quantifiers. The result is contained in the following theorem and its proof.

T25. Every definite formula without quantifiers is admissible.

Proof. Let $\underline{t}$ be definite without quantifiers. If $\underline{t}$ is of degree zero, then the theorem is true by T12. Therefore, we take $\underline{t}$ to be of degree $m > 0$.

Let $\underline{t}$ involve k elementary formulas $\underline{f}_1, \ldots, \underline{f}_k$. The first step is to write $\underline{t}$ in its (logically equivalent) "distinguished" disjunctive normal form [9]. This is a uniquely determined disjunction obtained by a mechanical procedure and having the property:

A. Each component of the disjunction (called a disjunct) is a conjunction of k components,[†] the i^{th} one being either $\underline{f}_i$ or the negation of $\underline{f}_i$. No two of these disjuncts are the same and none are contradictory.[††]

B. Thus, the free variable sequence of each disjunct is the same as the free variable sequence of the entire disjunction.

C. Furthermore, by parts A and B, the value set of $\underline{t}$ (or $\underline{t}$ preceded by an appropriate permutation operator) is given by the disjoint union of the value sets of the disjuncts.

D. It now follows that each disjunct is definite. To show this consider a disjunct $\underline{t}'$. Let $\underline{r}$ represent a typical unnegated elementary component of $\underline{t}'$ and let $\underline{s}$ represent a typical elementary negated component. Let $\underline{b}^m \in \omega(\underline{t}')$. Then, by applying the substitution $[\omega(\underline{t}') \rightarrow \underline{b}^m]$ to $\underline{t}'$, we obtain substitution instances $\underline{r}_i$ and $\underline{s}_j$ for $\underline{r}$ and $\underline{s}$, respectively, and for which

$$\omega(\underline{r}_i) = 1, \ \omega(\underline{s}_j) = 0$$

By T8 we have

$$\omega_*(\underline{r}_i) = 1, \ \omega_*(\underline{s}_j) = 0$$

[†] We use the well-known associative properties of ØR and AND. Thus we can speak of a disjunction or a conjunction of any number of components.

[††] If the entire formula is contradictory, then the mechanical procedure for generating the disjunctive normal form will yield no disjuncts.

and so $\underline{b}^m \in \omega_*(\underline{t}')$. Now let $\underline{b}^m \in \omega_*(\underline{t}')$. Then we can reverse the above argument provided the induced substitutions $\underline{r}_i$ and $\underline{s}_j$ are permissible formulas of the data base (i.e., do not contain '*'). But this is the case, since $\underline{b}^m \in \omega_*(\underline{t}')$ gives $\underline{b}^m \in \omega_*(\underline{t})$, which implies, by the definitude of $\underline{t}$, that $\underline{b}^m$ cannot contain '*'. Hence, part D.

E. Consider now each disjunct $\underline{t}'$ as consisting of two parts: the conjunction of the unnegated components (call this 'R'), and the conjunction of the remaining components (call this 'S'). We will show that the free variable set of S is included in the free variable set of R.

First, by part A, the disjunct is not contradictory and, by hypothesis, it is of degree $m > 0$. Now suppose there were free variables in S not in R. Then, by D13a, we can choose a data base in which the value set of the disjunct is not null. Thus, there is a $\underline{b}_1^m$ for which we have

$$\omega([\varphi(\underline{t}') \rightarrow \underline{b}_1^m](\underline{t}')) = 1 \qquad (4)$$

Next, we modify the substitution schema in (4) by sending each variable of $\bar{\varphi}(t')$ which is not in R into '*', while leaving the other substitutions unchanged. This gives a substitution schema

$$[\varphi(\underline{t}') \rightarrow \underline{b}_2^m] \qquad (5)$$

where $\underline{b}_2^m$ contains '*'. Again let $\underline{r}$ be a typical component of R and let $\underline{s}$ by a typical elementary negated component

of S. Then the modified substitution schema (5) will give induced substitution instances for which

$$\omega_*(\underline{r}_i) = 1, \ \omega_*(\underline{s}_j) = 0$$

Hence,

$$\omega_*([\varphi(\underline{t}') \rightarrow \underline{b}_2^m](\underline{t}')) = 1$$

and $\underline{b}_2^m$ is in $\omega_*(\underline{t}')$. But this contradicts the fact that $\underline{t}'$ is definite. Therefore, $\bar{\varphi}(S) \subset \bar{\varphi}(R)$.

F. To put each disjunct in proper form, we first take the conjunction R of the unnegated components. By T23a, R is proper. Let $\underline{f}_{i_1}, \underline{f}_{i_2}, \ldots, \underline{f}_{i_n}$ be the sequence of negated elementary components remaining. From part E we have for each of these

$$\bar{\varphi}(\underline{f}_{i_j}) \subset \bar{\varphi}(S) \subset \bar{\varphi}(R) \qquad (6)$$

Consequently, by T23b, (6), and the propriety of R, the formula

$$(\ldots((R)\mathrm{BN}(\underline{f}_{i_1}))\mathrm{BN}\ldots)\mathrm{BN}(\underline{f}_{i_n}) \qquad (7)$$

is proper.

After putting each disjunct in the form (7), we then write their disjunction. By T23c and part B, this will be a proper formula. The transformation is completed by prefixing the operator $(P\varphi(\underline{t}))$ to the result (T23e). Thus $\underline{t}$ is admissible by D16.

This proves T25.

5.4 ON OTHER BINARY OPERATORS

We have included the operator BN in the class β because the proper formulas then become a useful tool in the evaluation of definite formulas. The result of the previous section (5.3) gives an example of this.

The class β includes five operators. Three others could be considered. Would their use effect the class of admissible formulas? We examine each operator with this question in mind.

1. The Sheffer Stroke (SS). This gives the negation of conjunction. A formula with this operator and having definite components r and s will be definite if and only if the formula is of degree zero. We can write (r)SS(s) in the logically equivalent form

$$\text{NØT}((\underline{r})\text{AND}(\underline{s})) \tag{8}$$

or, using binary operators,

$$(\text{T})\text{BN}((\underline{r})\text{AND}(\underline{s})) \tag{9}$$

Thus, if (r)SS(s) is definite with proper components, then r and s are of degree zero, and so (8) and (9) are proper (T23).

2. The exclusive 'or' (EØR). This gives the negation of equivalence. For a formula with this operator we have a theorem analogous to T16. That is, if r and s are definite and neither r nor s is contradictory, then: (r)EØR(s) is definite if and only if the free variable

sets of $\underline{r}$ and $\underline{s}$ are the same. For $(\underline{r})E\emptyset R(\underline{s})$ we use the logically equivalent form

$$((\underline{r})\emptyset R(\underline{s}))BN((\underline{r})AND(\underline{s})) \qquad (10)$$

Consequently, if $(\underline{r})E\emptyset R(\underline{s})$ is definite with proper components, neither of which is contradictory, then, by the above theorem and T23, (10) is proper.

3. Neither ... nor (NØR). This gives the negation of disjunction. For a formula with this operator we have a theorem analogous to T19. If $\underline{r}$ and $\underline{s}$ are definite and neither are tautologies, then: $(\underline{r})N\emptyset R(\underline{s})$ is definite if and only if the free variable set is null. We can write $(\underline{r})N\emptyset R(\underline{s})$ in the logically equivalent form

$$N\emptyset T((\underline{r})\emptyset R(\underline{s})) \qquad (11)$$

or, alternatively,

$$(T)BN((\underline{r})\emptyset R(\underline{s})) \qquad (12)$$

Thus, if $(\underline{r})N\emptyset R(\underline{s})$ is definite with proper components neither of which is a tautology, then, by the above theorem and T23, (11) and (12) are proper. These three analyses therefore show that the class of admissible formulas is not enlarged by introducing these operators.

5.5 UNIVERSAL FORMULAS

We now come to the more difficult questions of admissibility. These deal with definite, but improper, formulas with quantifiers.

Implications. Consider first universal quantification over an implication. A basic situation is given by the formula

$$(Ay)(Fy \text{ IMP } Gy) \tag{13}$$

This states, in the object language, that the property F is subsumed under the property G. In Sec. 2 we gave an example of a natural-language question which led to a formula of this kind.† The admissibility of (13) is covered by the following theorem:

T26. If $\underline{s}$ and $\underline{t}$ are proper, and $\varphi(\underline{s}) = \varphi(\underline{t}) = \underline{v}$, then

$$(A\underline{v})((\underline{s})\text{IMP}(\underline{t})) \tag{14}$$

is definite and admissible.

Proof. (14) is logically equivalent to

$$N\text{Ø}T((E\underline{v})((\underline{s})BN(\underline{t}))) \tag{15}$$

But this is proper by T23.

This proves T26.

Now (13) is a closed sentential formula. An example question which leads to an open sentential formula is

Who wrote all the papers on
question-answering systems? (16)

†See p. 12, formulas (7), (8).

A possibly acceptable symbolic translation of this is the formula

$$(Ay)(Fy \text{ IMP } Wxy) \qquad (17)$$

where 'F' is an abbreviation for a complex predicate expression denoting the property of being a paper on question-answering systems.

We have not been primarily concerned with the method of symbolization;† but, in the case of (16), the method becomes important. First we replace 'who' in (16) by 'x'. Next we replace 'wrote' by 'W'. Finally, we replace the phrase 'the papers on question-answering systems' by 'F', where 'F' abbreviates a complex predicate expression which we suppose has been generated. The question (16) is now in the form 'x W all F' or

$$Wx(\text{all } F) \qquad (18)$$

Thus, (18) is in two-place predicate form with an argument-like expression 'all F' occupying the second position. This expression is indeed an argument. It is a restricted quantified variable [3, §30]. This means that the values of the variable are restricted to a certain class; in our example, the class F. We write 'y^F' for the restricted variable, and have the symbolization rule

'$(Ay^F) \ldots y^F \ldots$' for '... all F ...'

---

†See p. 5, footnote.

(The series of dots represents the context of 'all F'.) Using this rule, (18) then becomes

$$(Ay^F)(Wxy^F) \tag{19}$$

The definition of restricted quantifiers now comes into play. '(Ay^F)' is defined by:

$$'(Ay^F) \ldots y^F \ldots' \text{ for } '(Ay)(Fy \text{ IMP } \ldots y \ldots)' \tag{20}$$

and '(Ey^F)' is defined by:

$$'(Ey^F) \ldots y^F \ldots' \text{ for } '(Ey)(Fy \text{ AND } \ldots y \ldots)' \tag{21}$$

Now all of the logical equivalences of the predicate calculus hold for restricted quantifiers providing the class F is not empty [3, §30]. Thus, by applying the definition (20) to (19), we obtain (17) plus an additional (implicit) meaning component; namely, the class F is not empty. The important thing here is that the symbolization which most closely matches the natural-language structure exposes this meaning component for us.†

---

†Similarly, the example question of Sec. 2, having the form 'Are all F's G's?', becomes

$$(Ax^F)(Gx^F)$$

with the additional meaning component that F is not empty. The paraphrase 'Is every F a G?' receives a different initial symbolization, namely,

$$(Ax^F)(x^F = (\eta y)(Gy))$$

but has the same end result.

We will later return to this example and solve the admissibility problem for it. But first we will make a systematic study of universal implications. This will lead to a generalization of T26.

Consider an implication of degree greater than zero and having proper components. By T18, such a formula will be indefinite, and, by the comments following T21, its universal quantifications are of potential interest. They are quantification of a variable occurring 1) only in the implicans, 2) only in the implicate, or 3) in both. The three cases are modelled by:

$$(Ax)(Fxy \text{ IMP } Gyz) \tag{22}$$

$$(Az)(Fxy \text{ IMP } Gyz) \tag{23}$$

$$(Ay)(Fxy \text{ IMP } Gyz) \tag{24}$$

If any variables are absent, we get various subcases. (There are 12 of these plus an additional 12 if we include another free variable of the kind that is quantified.) For example, (13) and (17) are subcases of (24).

The formula (22) is logically equivalent to

$$(Ex)(Fxy) \text{ IMP } Gyz \tag{25}$$

By T23 this is again an implication having proper components (i.e., it has the model 'Fy IMP Gyz').

The formula (23) is logically equivalent to

$$Fxy \text{ IMP } (Az)(Gyz) \tag{26}$$

Now, the second component of (26) involves the universal quantification of a proper formula. Thus, because of T21, we will regard (23) as degenerate. That is, if the second component of (26) were definite, then it would be contradictory. Consequently, (26) would become the indefinite formula 'NØT(Fxy)'. We therefore define:

D17. $(Av)((s)IMP(t))$ is degenerate if $v \notin \bar{\varphi}(s)$.

The non-degenerate formulas (22) and (24) are then the ones of interest.

T27. Let u be a non-degenerate universal implication with proper components in the implication, and let r be any proper formula with $\bar{\varphi}(u) \subset \bar{\varphi}(r)$, then

$$(r)AND(u) \tag{27}$$

is definite and admissible.

Proof. Let u be the formula

$$(Av)((s)IMP(t))$$

First, we can assume without loss of generality that v is not in $\bar{\varphi}(r)$. For if it were, then we could write u in a logically equivalent form by replacing v with a variable not occurring in either $\bar{\varphi}(u)$ or $\bar{\varphi}(r)$. Formula (27) is then logically equivalent to the following formula (28):†

---

†Proof. We are to show that if v is not in $\bar{\varphi}(r)$, then (in traditional notation)

$$r.(Av)(s \supset t) \equiv r.\sim(Ev)(r.s.\sim t)$$

Now, since v is not in $\bar{\varphi}(r)$, the left side is equivalent to

$$(\underline{r})\mathrm{BN}((\mathrm{E}\underline{v})(((\underline{r})\mathrm{AND}(\underline{s}))\mathrm{BN}(\underline{t}))) \qquad (28)$$

Since $\underline{u}$ is non-degenerate and since $\bar{\varphi}(\underline{u}) \subset \bar{\varphi}(\underline{r})$, it follows that $\bar{\varphi}(\underline{t}) \subset \bar{\varphi}((\underline{r})\mathrm{AND}(\underline{s}))$. Hence, by T23a,b, the formula $((\underline{r})\mathrm{AND}(\underline{s}))\mathrm{BN}(\underline{t})$ is proper. Prefixing $(\mathrm{E}\underline{v})$ to this gives, again, a proper formula. Consequently, by T23b, (28) is proper.

This proves T27.

T26 is a corollary of T27, To show this we write (14) in the form

$$(\mathrm{T})\mathrm{AND}((\mathrm{A}\underline{v})((\underline{s})\mathrm{IMP}(\underline{t})))$$

T27 is now applicable to this under the conditions of T26.

An Admissibility Problem. Consider again formula (17). This formula is a non-degenerate universal implication. As it stands, it is at most semi-definite. That is, on a data base for which F is empty, the formula is indefinite. If these data bases are excluded by conjoining

---

$$(\mathrm{A}\underline{v})(\underline{r}.(\underline{s} \supset \underline{t}))$$

which is equivalent to

$$(\mathrm{A}\underline{v})(\underline{r}.(\underline{r}.\underline{s} \supset \underline{t}))$$

and, again by hypothesis, to

$$\underline{r}.(\mathrm{A}\underline{v})(\underline{r}.\underline{s} \supset \underline{t})$$

This last is, by eliminating '$(\mathrm{A}\underline{v})$', equivalent to

$$\underline{r}.\sim(\mathrm{E}\underline{v})(\sim(\underline{r}.\underline{s} \supset \underline{t}))$$

and hence to

$$\underline{r}.\sim(\mathrm{E}\underline{v})(\underline{r}.\underline{s}.\sim\underline{t})$$

This proves the equivalence of (27) with (28).

a second meaning component stating that F is not empty, then the formula becomes definite. This component is

$$(Ez)(Fz) \tag{29}$$

Thus we consider the formula

$$(Ez)(Fz) \text{ AND } (Ay)(Fy \text{ IMP } Wxy) \tag{30}$$

This, not (17), is the preferable symbolization of (16). Now T27 is not directly applicable to (30). We need a formula to fill the role of r. This is provided when we note that (30) implies that x is in the domain of W; i.e., (30) implies

$$(Ev)(Wxv)$$

Consequently, (30) is logically equivalent to

$$(Ez)(Fz) \text{ AND } ((Ev)(Wxv) \text{ AND } (Ay)(Fy \text{ IMP } Wxy)) \tag{31}$$

The second component of (31) now satisfies the conditions of T27.

To summarize these results:

We can generally handle a universal implication if it occurs in a context satisfying the requirement of T27. This includes the important special case of a closed universal implication. This second result is stated separately in T26. Certain exceptional cases of universal implications can be treated. These are conditionally admissible formulas. An example is given by (17).

Closed Universal Formulas. Having examined universal quantification over implication, let us now look at universal quantification over other indefinite formulas. We restrict the discussion to those having a single free variable and proper components. The quantification then closes the formula. This gives exactly eight formulas† to be considered.

1. (Ax)(NØT(Fx)). This has the proper equivalent

$$\text{NØT}((\text{Ex})(\text{Fx}))$$

In the object language this says that the property F does not occur.

2. (Ax)(p BN Fx). The proper equivalent is

$$\text{p BN } (\text{Ex})(\text{Fx})$$

This says: p, and the property F does not occur.

3. (Ax)(p ØR Fx). This is reducible to a degenerate implication; i.e., the component is equivalent to

$$(\text{NØT(p)}) \text{ IMP } (\text{Ax})(\text{Fx})$$

4. (Ax)(p IMP Fx). This is a degenerate implication--a special case of (23).

5. (Ax)(Fx IMP p). This is a special case of (22) and gives the proper equivalent

---

†We include the three implications again, so the list is complete.

(Ex)(Fx) IMP p

The meaning is: if the property F occurs, then p.

6. (Ax)(Fx IMP Gx). This is the basic situation covered by T26. The proper equivalent is

NØT((Ex)(Fx BN Gx))

In the object language this means that the property F is subsumed under the property G.

7. (Ax)(Fx IFF p). This is reducible to a conjunction of universal implications,† one of which is degenerate.

8. (Ax)(Fx IFF Gx). This is reducible to a conjunction of two universal implications both covered by formula 6.

The inventory shows again why formula 6 can be regarded as the fundamental use of the universal quantifier; i.e., if F and G are "natural" properties†† (hence represented by elementary formulas on a data base), then formulas 1 and 2 would be expected to be false and formula 5 would say the same as 'p'.

This concludes our limited study of universal formulas. The general question of the admissibility of definite formulas with quantifiers is still open.

---

†The component is first transformed to a conjunction of implications, then the universal quantifier is distributed over the conjunction.

††See Sec. 4.7, p. 87.

5.6 CONCLUDING REMARKS

Admissibility of Definite Formulas--A Conjecture. We conjecture that every definite formula is admissible. If this could be proved in any manner, it would be of theoretical interest; but if a proof were given that explicitly shows the transformation of a definite formula to a proper equivalent, it would be of great practical use. Such a direct proof was given for definite formulas without quantifiers (T25).

Definitude and "Natural" Formula. In Sec. 2 (p. 8) the notion of definite formula was introduced as an exact concept to replace the vague concept of reasonableness or acceptability of input queries to a computerized question-answering system. We would also like to say that a definite formula is somehow "natural."

In what sense do our theorems lend support to the view that definitude is an adequate explicatum?† Certain results are of interest here. T21 confirms our intuition on "natural" uses of universal quantifiers. The analyses of Sec. 4.7 dealing with proper formulas and existential quantifiers showed how these were related to certain common natural-language structures. In Sec. 5.5 less comprehensive studies were given on the definitude of universal formulas. Evidently, further results in this area would have value for machine processing of natural language.

---

†See Ref. 6 or Ref. 4 for a discussion of the general process of explication. Carnap calls the exact concept or its term the explicatum; the vague concept or its term, the explicandum.

Definitude--Theory II. We have proposed one definition of definitude. Is this the best one? During the investigation we considered a second, and weaker, definition. It was developed as follows:

First, define the constituents of $\underline{s}$ to be the elementary formulas that enter into it. Let $\omega_i(\underline{s})$ be the value set of the i^{th} constituent of $\underline{s}$. Next, define $\bar{\omega}(\underline{s})$ to be the set of $\underline{d}$'s which occur in members of $\omega(\underline{s})$; define $\bar{\bar{\omega}}(\underline{s})$ to be the union of the $\bar{\omega}_i(\underline{s})$, $i = 1, \ldots, n$ (n being the number of constituents of $\underline{s}$). Finally, we define $\underline{s}$ to be semi-definite with respect to a data base if and only if

$$\bar{\omega}(\underline{s}) \subset \bar{\bar{\omega}}(\underline{s})$$

for the data base. As before, $\underline{s}$ is definite if it is semi-definite for every data base.†

In other words, s is definite if the expressions that go to make up its value set are trapped amongst the expressions that go to make up the value sets of its constituents.

Under this definition every formula of degree zero is definite. Also a quantification of a definite formula is definite.

---

† In order to carry out this second theory, a postulate is required which states, in essence, that the universal property cannot be formulated in an elementary formula. This postulate then provides a $\underline{d}$ which plays the role of '*' in the proofs.

Certain technical difficulties prevented us from fully developing this second theory.[+] If this could be carried through, the resulting class of definite formulas would be of interest.

[+] A rather surprising road block appeared when we were unable to prove that a substitution instance of a definite formula is definite.

6. IMPLEMENTATION OF THE THEORY

6.1 INITIAL REMARKS

We have been motivated entirely by the possibility of computer implementation and our theory has been formulated with this major consideration. It remains only to tie these formulations together in a general flow of processing procedures. The following discussion provides some guidelines and suggestions for the implementation.

6.2 ANALYZING A FORMULA

Given an input formula, the first step is to set up a suitable worktable for subsequent processing. This routine is referred to as analyzing a formula.†

Input. This consists of 1) a sentential formula, and 2) a table of designs (shown in Table 4). Design 3 is for an elementary formula--characterized by an absence of parentheses.

Table 4

DESIGN TABLE

1	$\underline{\alpha}(\underline{s}_1)$
2	$(\underline{s}_1)\underline{\beta}(\underline{s}_2)$
3	$\underline{f}$

†Several methods are available here. The suggested method seems most appropriate for our definition of sentential formula.

Output. This consists of a table with four columns: Ind (for 'index'), Op (for 'operator'), Form (for 'formula'), Val (for 'value'). The entries in this table are defined as follows:

Ind_i = index (explanation to follow) of the i^{th} subformula

(By 'subformula' we mean any component, component of a component, etc., of the input formula. We also include the input formula itself as a subformula.)

Op_i = operator for the i^{th} subformula

(If the i^{th} subformula is elementary, then $Op_i = 0$.)

$Form_i$ = 0, if the i^{th} subformula is not elementary; otherwise, the subformula, or a pointer to the subformula

Val_i = 0

Val_i is a reserved space to store information on the i^{th} subformula--either the information itself or a pointer to that information (e.g., free variable sequences, value sets, propriety determinations, etc.)

Explanation of Index. The index is a numerical device allowing us to move up and down the "tree" of the formula. Exactly:

1) If $i = 1$, $Ind_i = 1$.

2) Let N be any integer greater than or equal to one, and so N is made up of digits

$$N = \gamma_1\gamma_2\cdots\gamma_n$$

with $\gamma_1 \neq 0$; we then define

$$(\gamma, N) = \gamma\gamma_1\gamma_2\cdots\gamma_n$$

For example, if $N = 21$, then $(2, N) = 221$.

3) If the i^{th} subformula has one component, then that component receives the index

$$Ind_j = (1, Ind_i)$$

If the i^{th} subformula has two components, then the two components receive the indices

$$Ind_j = (1, Ind_i)$$

$$Ind_k = (2, Ind_i)$$

Thus, given any Ind_i not equal to one, it will have the structure (γ, N) and so the i^{th} subformula will enter into the determination of the subformula with index N. Moreover, if Op_i is not equal to zero, then the i^{th} subformula will depend on one or two subformulas whose indices are as described above.

The analysis routine is now executed according to the charts shown in Fig. 3.

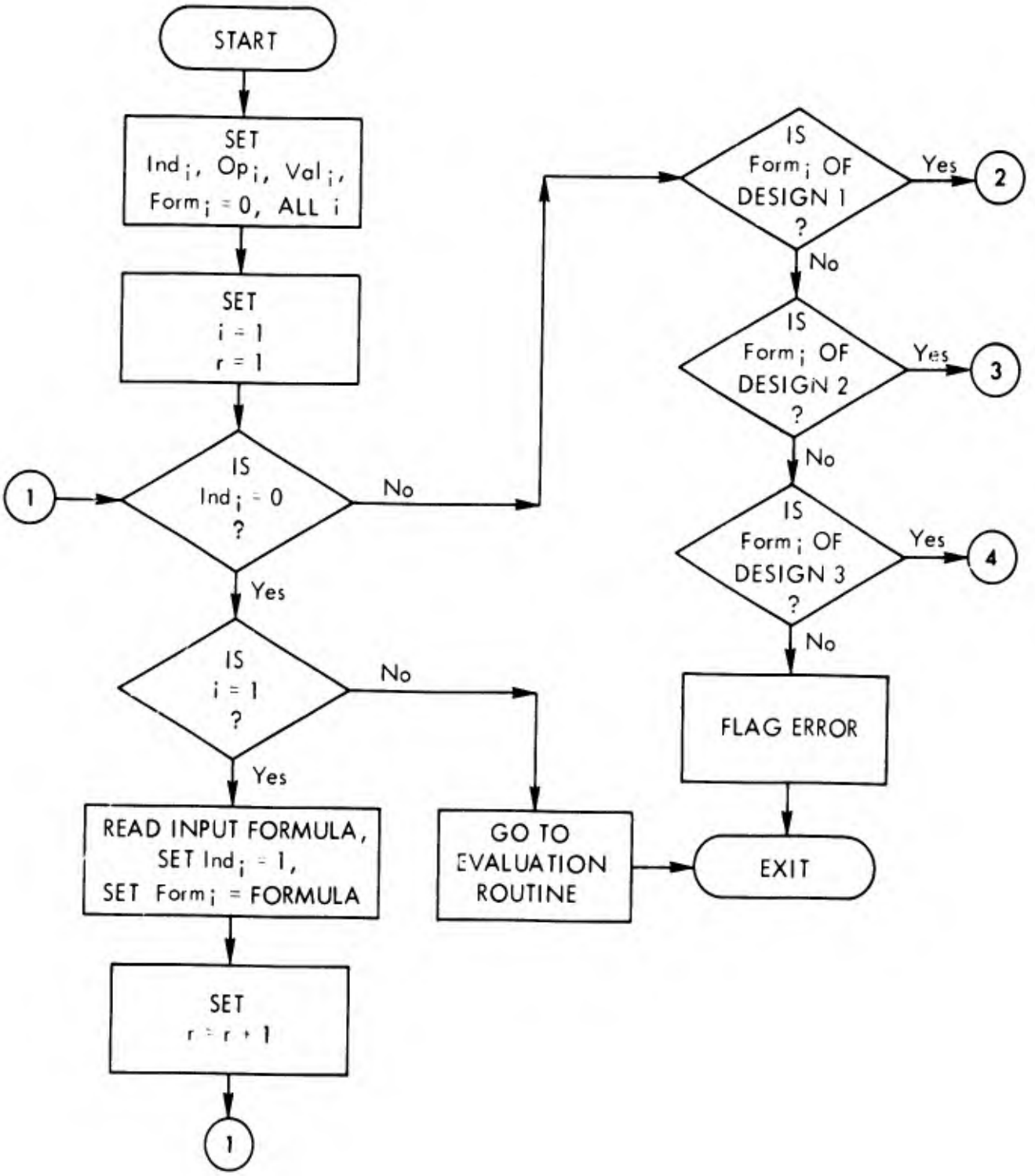


Fig. 3A--Chart for Analysis Routine

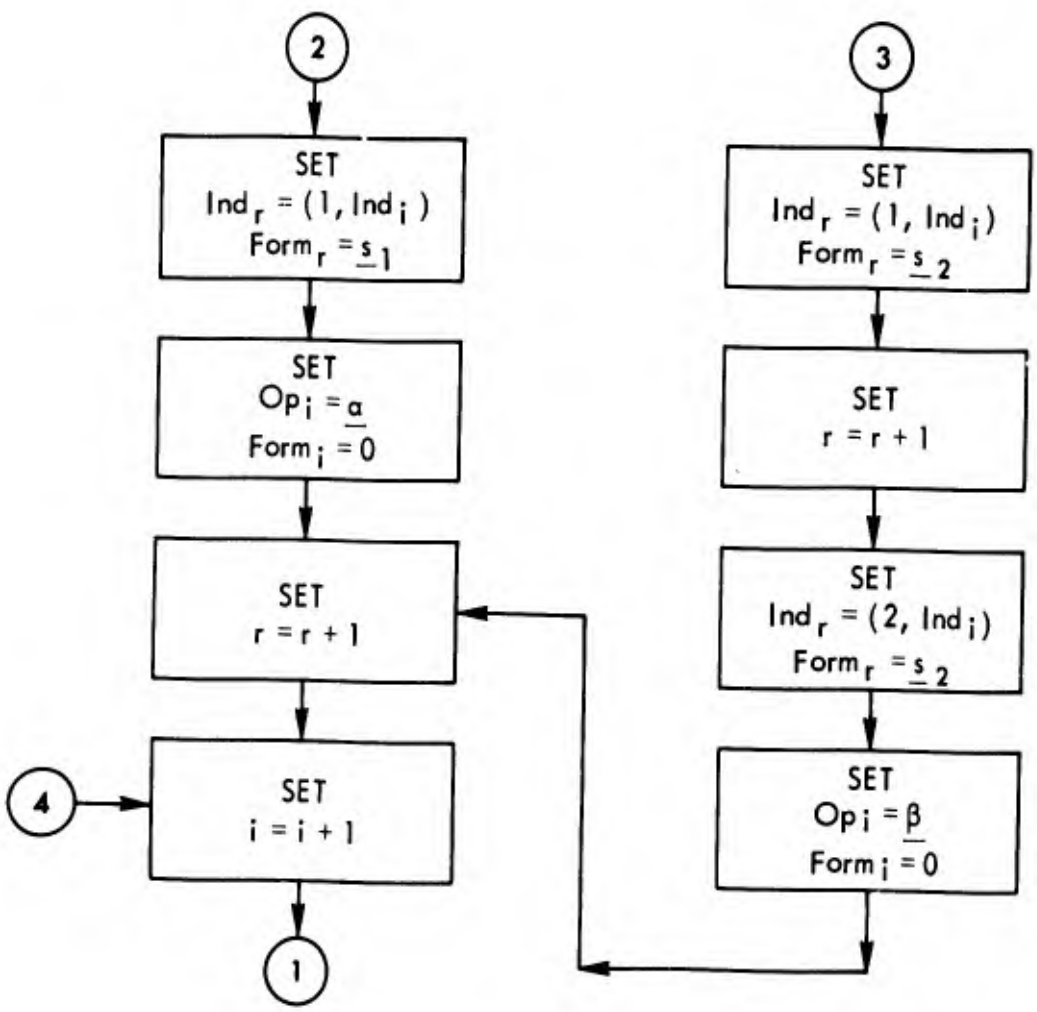


Fig. 3B--Chart for Analysis Routine (concluded)

Example: Consider the sentential formula

$$(Ey)((B,y)AND(W,x,y)) \qquad (1)$$

(Under the interpretation of Table 1, p. 4, this corresponds to the natural-language question: 'Who has written a book?'.) Given the input (1), the result of the analysis routine is shown in Table 5. The entries in the Val column are all 0 after this processing.

Table 5

ANALYSIS OF INPUT FORMULA (1)

i	Ind	Op	Form	Val
1	1	(Ey)	0	0
2	11	AND	0	0
3	111	0	B,y	0
4	211	0	W,x,y	0

6.3 THE ψ-ROUTINES

Using the output of the analysis routine, we can evaluate the formula by beginning with the elementary formulas and working up the "tree". This requires a succession of evaluations of various types on the subformulas--each evaluation dependent upon prior evaluations. Called ψ-routines, these enter into a general evaluation routine to be described in Sec. 6.4.

The sequence of evaluations is shown in Fig. 4. Each numbered box corresponds to an execution of the general evaluation routine with a different specification for the ψ-function. Thus:

Step 1. Set $\psi = \varphi$. This routine determines the free variable sequence of each subformula according to the definition D5.

Step 2. Set ψ = PC. This is the propriety check. If a subformula is proper, we define PC = 1; otherwise, PC = 0. The input to this routine consists of the free variable determinations from Step 1. The rules of the characterization theorem (T23) then define the routine.

Step 3. Set ψ = AVS. This routine uses the results of Step 1 to calculate the auxiliary variable sequences X, Y, Z for the binary subformulas according to D14.

The next step is to identify which rules of Sec. 4.6 (regarded as subroutines) are required for the calculation of value sets and draw them into the program. We call this the rule determination RD.

Step 4. Set ψ = RD. If the operator of the i^{th} subformula is singulary, then the value is one of the rule numbers: 1, 2, or 3. If the operator is binary, then the value is one of the rule numbers 4-19; this is determined by the classification given by the auxiliary variable sequence in Table 3 (p. 73). The value RD = 0 corresponds to the procedure for elementary formulas.

Step 5. Set $\psi = \omega$. The value set of the formula is then calculated according to the assembled subroutines (Rules 1-19, Sec. 4.6).

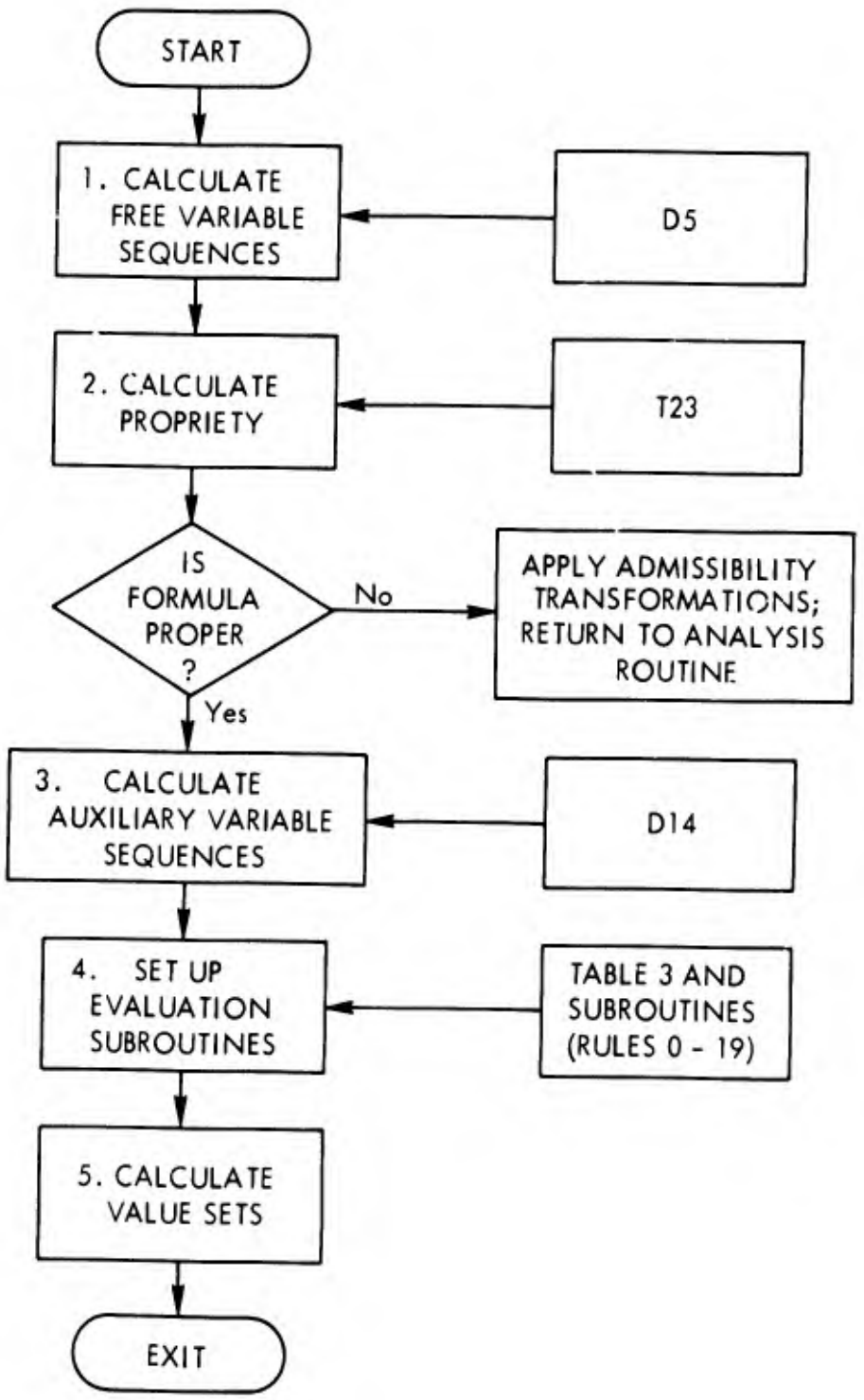


Fig. 4--Sequence of ψ-Routines

6.4 EVALUATING A FORMULA

Each of the υ-routines has the common feature of depending only on the operator of a subformula and the ψ-values of the components. Thus, we give a general schema for evaluating a formula (see Fig. 5).

Input. This consists of 1) the table resulting from the analysis routine, and 2) a specification of ψ.

Output. This consists of the input table with

$$Val_i = \psi(i)$$

Thus, at the final execution of the sequence of evaluations (Steps 1-5, Fig. 4), the entry for Val_1 with $\upsilon = \omega$ is the value set of the formula.

Example: The evaluation of formula (1) is shown in Table 6. The evaluation uses the example data base of Table 2 (p. 4).

Table 6

EVALUATION OF INPUT FORMULA (1)

i	Ind	Op	Form	φ	PC	X	Y	Z	RD	ω	L(i)
1	1	(Ey)	0	x	1	0	0	0	1	Array 4	1
2	11	AND	0	y, x	1		y	x	6	Array 3	0
3	111	0	B, y	y	1	0	0	0	0	Array 2	0
4	211	0	W, x, y	x, y	1	0	0	0	0	Array 1	0

Array 1	
x	y
a	b
a	c
a	d
e	f
g	h
i	h

Array 2
y
b
d
f
h

Array 3	
x	y
b	a
d	a
f	e
h	g
h	i

Array 4
x
a
e
g
i

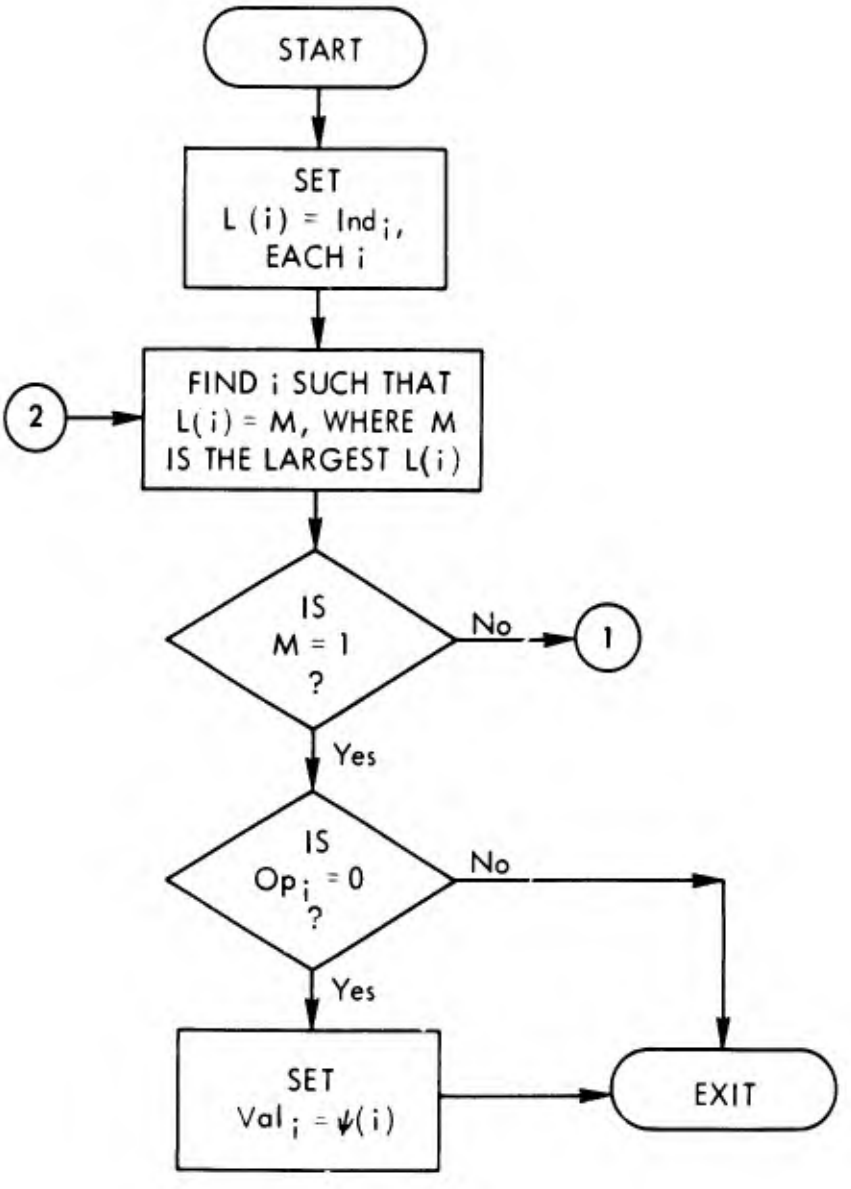


Fig. 5A--Chart for General Evaluation Routine

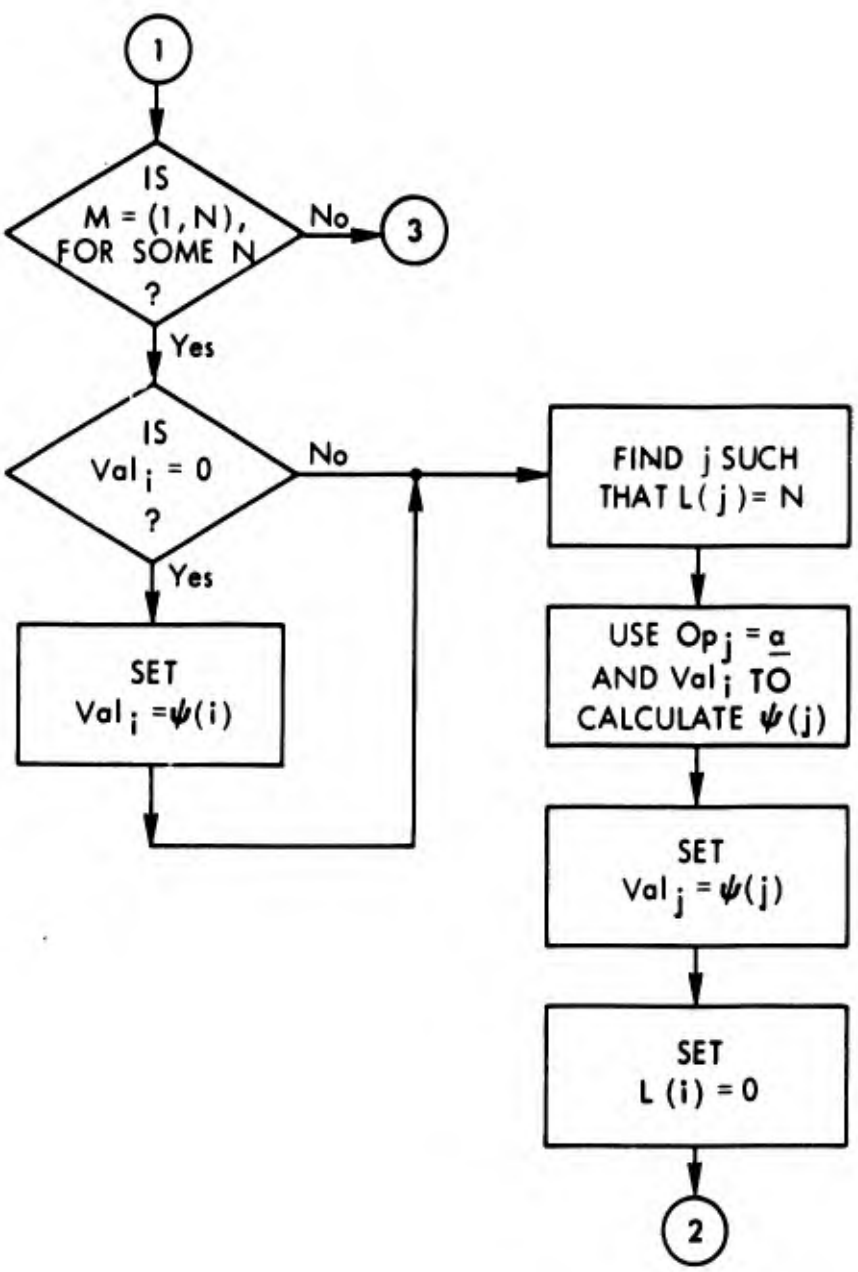


Fig. 5B--Chart for General Evaluation Routine (continued)

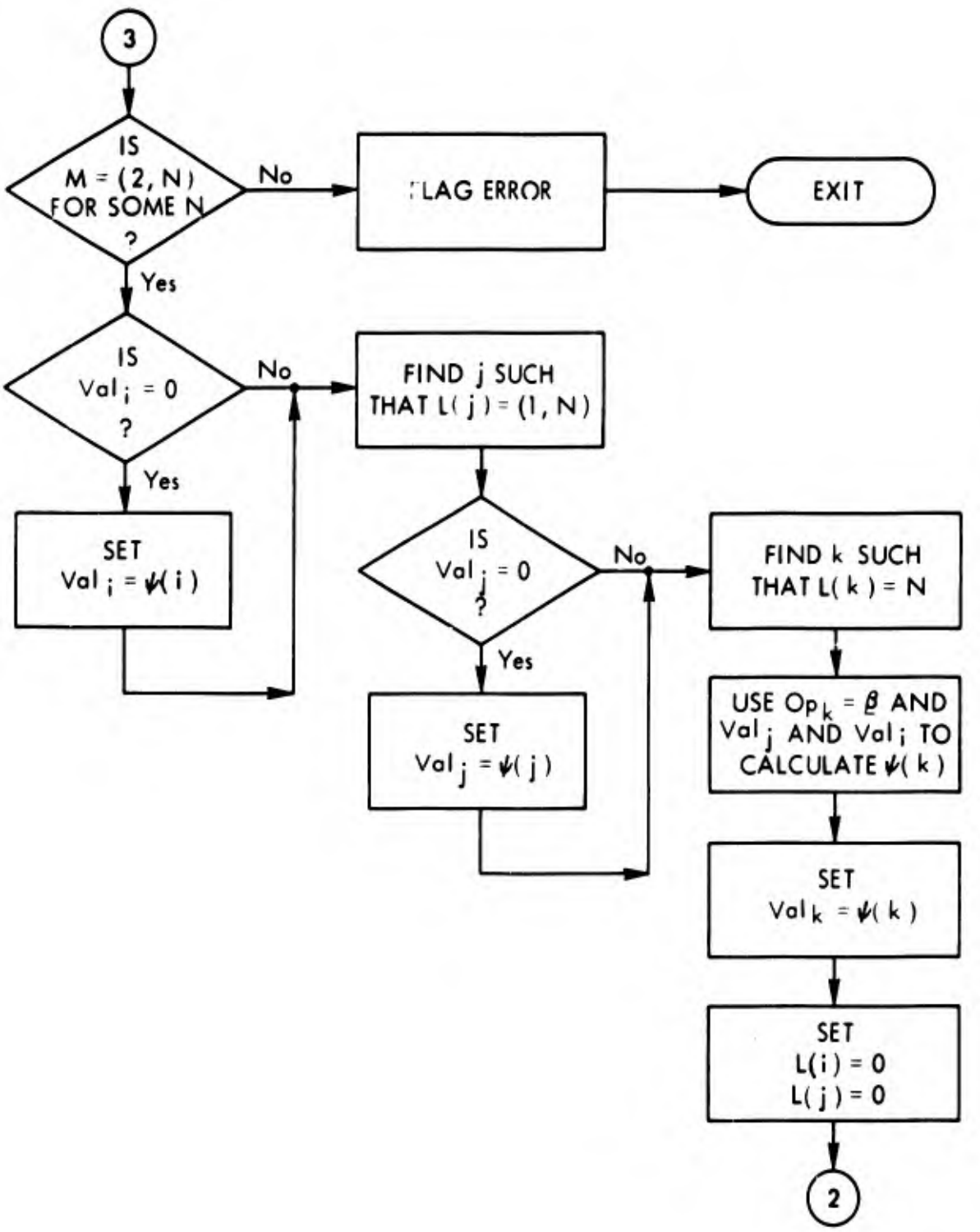


Fig. 5C--Chart for General Evaluation Routine (concluded)

6.5 ADMISSIBILITY TRANSFORMATIONS

The following discussion is restricted to the case of an improper formula without quantifiers.† By T25, the definitude of such a formula rests entirely on the propriety of its "distinguished" disjunctive normal form.

The suggested method for obtaining this normal form builds upon the routines described above and uses arithmetic computations rather than logical transformations. It should be satisfactory in all practical situations. By this we mean that the number of different elementary formulas is small (say, not exceeding six or seven), and so the execution time on a large computer will be about two seconds.

The input to the procedure is the output of the analysis routine (Sec. 6.2); from this a new analysis table is to be constructed.

Step. 1. Identify the different elementary formulas in the analysis table (i.e., the distinct formulas with $\text{Form}_i \neq 0$). Let there be m of these: $\underline{f}_1, \ldots, \underline{f}_m$.

We now use the following fact:

A disjunct in which $\underline{f}_{i_1}, \ldots, \underline{f}_{i_p}$ are negated and $\underline{f}_{i_{p+1}}, \ldots, \underline{f}_{i_m}$ are unnegated will appear in the disjunctive normal form if and only if the evaluations

---

†See Sec. 5.3, pp. 92-95. For the case of an improper formula with quantifiers our techniques are limited to 1) rewriting the input formula according to T26 or T27, or 2) distributing quantifiers and testing the resulting formula for propriety.

$$\omega(\underline{f}_{i_1}) = \ldots = \omega(\underline{f}_{i_p}) = 0 \qquad (2a)$$

$$\omega(\underline{f}_{i_{p+1}}) = \ldots = \omega(\underline{f}_{i_m}) = 1 \qquad (2b)$$

give a value of 1 for the entire formula.

Step 2. Set up all possible evaluation schemata of the kind given by (2). There are 2^m of these. (Hence the limitation on the procedure.) Enter these in the Val column of the analysis table and evaluate the formula with $\psi = \omega$. Those results producing $Val_1 = 1$ give the disjuncts of the normal form. We suppose these results are assembled in an array of n rows (each row corresponding to a disjunct) and m + 1 columns. In each row, the entries are the p negated elementary formulas (taken first) followed by the remaining m-p unnegated elementary formulas. The last position gives the value of p. Let this array be F(i,j). For example, for the elementary formulas $\underline{f}_1$, $\underline{f}_2$, $\underline{f}_3$, the row

$\underline{f}_2$	$\underline{f}_1$	$\underline{f}_3$	1

represents the disjunct

$$((\underline{f}_1)\text{AND}(\underline{f}_3))\text{BN}(\underline{f}_2)$$

Step. 3. If p = m for any disjunct, the formula is indefinite. If n = 0, the formula is contradictory and its value set is null. In either case, the problem is

done. Suppose now that $n > 0$ and for every p, $0 \leq p < m$. We then proceed with the construction of a new analysis table.

Step 4. Set

$$Ind_1 = 1$$

For $i = 2, \ldots, n - 1$, set

$$Ind_i = (1, Ind_{i-1})$$

For $i = 1, \ldots, n - 1$, set

$$Op_i = \emptyset R$$

A counter r is now defined. It records how many rows have been filled; it is updated at various times. Set

$$r = n - 1$$

Step. 5. Let k be the number of disjuncts that have been processed. Thus, $k = 0, \ldots, n - 1$. To process the $(k + 1)^{th}$ disjunct, define first: For $k < n - 1$,

$$N = (2, Ind_{k+1})$$

For $k = n - 1$,

$$N = (1, Ind_k)$$

Then set

$$Ind_{r+1} = N$$

For i = 2, ..., m - 1, set

$$Ind_{r+1} = (1, Ind_{r+i-1})$$

For i = 1, ..., p, set

$$Op_{r+i} = BN$$

For i = p + 1, ..., m - 1, set

$$Op_{r+i} = AND$$

Step. 6. For each value $\ell = 0, \ldots, m - 1$, do the following (ℓ is the number of conjuncts that have been processed in the $(k + 1)^{th}$ disjunct):

Define, analogous to N, the index M:

For $\ell < m - 1$,

$$M = (2, Ind_{r+\ell+1})$$

For $\ell = m - 1$,

$$M = (1, Ind_{r+\ell})$$

Then set,

$$\text{Ind}_{r+m+\ell} = M$$

$$\text{Form}_{r+m+\ell} = F(k + 1, \ell + 1)$$

i.e., the $(\ell + 1)^{th}$ elementary formula in the $(k + 1)^{th}$ disjunct.

Step. 7. Reset r and return to Step 5.

This ends the procedure. The resulting table will be the analysis table for the disjunctive normal form.

6.6 CONCLUDING REMARK

We have completed one part of the study of answering questions by computer. A certain structure was assumed for the input inquiry. This assumption is justified by the fact that an inquirer may phrase an inquiry in this form [1] and that it is possible for a computer to cast a variety of natural-language questions into such a "strict" form. In a forthcoming Memorandum we shall discuss how this can be done.

LIST OF DEFINITIONS

A

DESCRIPTIVE LIST OF THEOREMS

B

REFERENCES

1. Levien, R., and M. E. Maron, Relational Data File: A Tool for Mechanized Inference Execution and Data Retrieval, The RAND Corporation, RM-4793-PR (DDC No. AD625409), December 1965.

2. Carnap, R., The Logical Syntax of Language (English ed.), Routledge and Kegan Paul Ltd., London, 1937.

3. Reichenbach, H., Elements of Symbolic Logic, MacMillan Company, New York, 1947.

4. Carnap, R., Meaning and Necessity, 2d ed., University of Chicago Press, Chicago, 1956.

5. Reichenbach, H., The Theory of Probability, 2d ed., University of California Press, Berkeley and Los Angeles, 1949.

6. Carnap, R., Logical Foundations of Probability, University of Chicago Press, Chicago, 1950.

7. -----, Introduction to Symbolic Logic and Its Applications, Dover Publications, Inc., New York, 1958.

8. Whitehead, A. N., and B. Russell, Principia Mathematica, 2d ed., Cambridge University Press, London, 1960.

9. Hilbert, D., and W. Ackermann, Principles of Mathematical Logic (English ed.), Chelsea Publishing Company, New York, 1950.

DOCUMENT CONTROL DATA

1 ORIGINATING ACTIVITY	2a. REPORT SECURITY CLASSIFICATION
THE RAND CORPORATION	UNCLASSIFIED
	2b. GROUP

3. REPORT TITLE

ANSWERING QUESTIONS BY COMPUTER: A LOGICAL STUDY

4. AUTHOR(S) (Last name, first name, initial)

Kuhns, J. L.

5. REPORT DATE	6a. TOTAL No. OF PAGES	6b. No. OF REFS.
December ~~1968~~ 1967	137	9

7. CONTRACT OR GRANT No.	8. ORIGINATOR'S REPORT No.
F44620-67-C-0045	RM-5428-PR

9a. AVAILABILITY / LIMITATION NOTICES	9b. SPONSORING AGENCY
DDC-1	United States Air Force Project RAND

10. ABSTRACT

A study of the processing of questions input to a computerized question-answering system such as the RAND Relational Data File (see RM-5085-PR). The process consists of (1) transforming the natural-language question into a symbolic question (i.e., a certain formula of predicate calculus) and (2) generating the answer by calculating the value set of the resulting formula. This study is addressed to the second step. A key problem is the identification of "reasonable" input queries. These are characterized by introducing the concept of definite formula. A particular class of definite formulas--the proper formulas--is especially suitable for machine processing. A set of machine-recognizable sufficient conditions for their identification is given, together with rules for calculating their value sets. The definite, but improper, formulas are also studied. It is shown that definite formulas without quantifiers can be transformed into proper equivalents. For definite formulas with quantifiers, limited but useful results are obtained.

11. KEY WORDS

Mathematical logic
Relational data file
Information retrieval