Introduction

Geoghegan's paper represents an original synthesis of some of the most recent results in ethnography and cognitive psychology. He presents an axiomatic theory of coding behavior in finite-dimensional cognitive domains. Among the advantages of Geoghegan's approach are the following:

1. In contrast to many formal theories of psychological process, this approach contains nothing counterintuitive. On the contrary, Geoghegan's results stem from intuitively clear primitives and highly plausible axioms. The outcome is a theory encompassing a precise synthesis of empirical results from both ethnography and cognitive psychology.

2. Substantively, Geoghegan's theory deals not only with the formal ("code") aspects of information processing systems but also with some of the real-time aspects of the cognitive processes involved. In the terminology current in linguistics and psychology, this theory makes a considerable advance in embedding an account of the subject's competence in a wider theory that includes noncompetence aspects of performance. In currently fashionable anthropological terminology, Geoghegan's formulation comes squarely to grips with the problems of so-called "psychological reality."

3. Geoghegan's method presents a distinct advantage over existing paradigmatic and taxonomic methods for representing cognitive/semantic domains (cf. Kay, P. 1966, *Current Anthropology* 7: 20–23) in that it presents a natural and explicit mechanism for expressing the relations among different domains. Geoghegan's recoding procedure, which allows the total formal structure of one domain (ordered rule) to serve as an item in the structure of another domain (assessment in another ordered rule), goes a long way toward solving the ethnographic problem of interrelation of semantic domains. Moreover, this achievement is attained by a formal device independently motivated by an impressive number of experimental findings in cognitive psychology [cf. Miller, G. A. 1956, *Psychological Review* 63: 81–97 and Miller, G. A., E. Galanter, and K. H. Pribram, 1960, *Plans and the Structure of Behavior* (New York: Holt, Rinehart and Winston)].

4. Formally, the paper can serve as an introduction to axiomatic method for anthropologists as well as an introduction to some of the basic concepts of naïve set theory and the theory of order relations. The endnotes contain sufficient information on mathematical prerequisites to enable a reader with two years of high school mathematics to follow the argument—although some readers who have had no contact with mathematics since the tenth grade may find it somewhat slow going.

Information Processing Systems in Culture

WILLIAM H. GEOGHEGAN

Introduction

Among the recent developments in anthropological theory, especially in ethnoscience and the more psychologically biased variety of componential analysis, there has been a tendency to look upon the production of socially conditioned activity as the end result of a series of information processing operations performed by individual native actors. This orientation has been characterized by an interest in discovering and formulating sets of "rules" which account for culturally appropriate acts in terms of the situations that properly evoke them. Componential analysis, for example, can be conceptualized as a partially deductive, partially inductive technique used for discovering a rule that maps sets of distinctive features onto members of a complete set of lexemes at one taxonomic level. Insofar as the elements and structure of such a rule are held to be cognitively valid, it is felt that the rule constitutes an adequate model for certain cognitive processes as well as for the structure of a given semantic domain and the associated overt linguistic behavior. A rule of this kind exemplifies the class of models for one possible theory of information processing systems.

Speaking in more general terms, Frake (1964a) introduced his notion of the "cultural code" with the following remark: "The ethnographer ... seeks to describe an infinite set of variable messages as manifestations of a finite shared code, the code being a set of rules for the socially appropriate construction and interpretation of messages [socially interpretable acts and artifacts]" (1964a: 132). He characterizes such rules as follows:

> If we want to account for behavior by relating it to the conditions under which it normally occurs, we require procedures for discovering what people are attending to, what information they are processing, when they reach decisions which lead to culturally appropriate behavior . . . it is not the ethnographer's task to predict behavior per se, but rather to state rules of culturally appropriate behavior. . . . The model of an ethnographic statement is not: "if a person is confronted with stimulus X, he will do Y," but "if a person is in situation X, performance Y will be judged appropriate by native actors." (1964a: 133)

To describe a culture, then, is not to recount the events of a society but to specify what one must know to make those events maximally probable. The problem is not to state what someone did but to specify the conditions under which it is culturally

This paper was originally presented in a somewhat simplified form at the Symposium on Mathematical Anthropology, Annual Meetings of the American Association for the Advancement of Science, Berkeley, California, on December 29, 1965. Since the preparation of the present version, research in native information processing systems has continued and has suggested a number of changes in the mode of presentation of the formal theory, in the method of analysis, and in the presentation of results. The suggested revisions, however, do not involve any major changes in the substantive content of the theory as expressed in this paper, and it has not, therefore, been altered since preparation. Forthcoming publications of subsequent research will include such recent revisions. I am indebted to H. C. Conklin, Roy D'Andrade, Charles Frake, Paul Kay, and A. Kimball Romney for many valuable criticisms and suggestions. Though none of these people necessarily agree with everything that is said here, the paper has benefited immeasurably from their assistance. Financial assistance for the author while this work was in progress was supplied by the National Science Foundation in the form of an NSF Cooperative Graduate Fellowship.

appropriate to anticipate that he, or persons occupying his role, will render an equivalent performance. (1964b: 112)

If the ethnographer's task is to describe the "code" of a particular culture in terms of its component rules, and if this description is to specify the content and structure of rules as they are actually used by native actors, then it is only logical to demand that we first develop some conceptualization of what naturally occurring IP (information processing) systems are like. That is to say, we should have in hand a theory that indicates the necessary and sufficient conditions for any logical entity (e.g., a particular ethnographic statement) to be a model for a naturally occurring information processing rule.

Although Frake's paradigm for an ethnographic statement ("if a person is in situation *X*, performance *Y* will be judged appropriate by native actors") does give us a starting point for such a theory, it leaves a number of relevant questions unanswered. Are we to postulate, for example, the existence of a different rule for each recognized situation and the action appropriate to it; or does a rule account for a set of contrasting acts, each of which is appropriate to one or more members of some *set* of situations? How are we to characterize a "situation" itself? Is it some kind of unitary indivisible phenomenon, or can it be factored into a number of discrete, and at least semi-independent, components? Is the result of applying an information processing rule always a "performance" (which seems to imply overt activity), or can it also be the receipt of additional information for a native actor's "internal" use? What is the nature of the information processing phenomenon itself; how is the information contained in a native actor's characterization of a particular situation operated upon to yield inferences concerning the appropriate output? And, finally, given the possibility that such rules can differ from one another in their degree of complexity, we can ask whether or not there exist any natural boundary conditions on the complexity of naturally occurring rules, arising, perhaps, from limitations on human information processing capabilities. These, and a number of related questions that will arise during the subsequent discussion, should be answerable in terms of an adequate theory of natural (or "cultural") IP systems.

It should be clear that the kind of phenomenon we are dealing with is cognitive as well as cultural, that information processing systems comprise part of the basic "mental apparatus" of individual native actors, and therefore that the elements of an adequate theory must represent classes of phenomena actually present in their cognitive maps. If all we desire is to account for the relationship between situations and performances as *we* define and conceptualize them, then there are any number of adequate theories (and models for these theories) which could be used to accomplish the task. Such an approach would go a long way toward summarizing data from the ethnographic record (and even teach us how to behave more or less unobtrusively within a given society), but it would not tell us very much about how the native actors themselves make culturally appropriate decisions. If, on the other hand, the adequacy of ethnographic description is to turn on whether or not it accounts for not only *what* a native actor does under certain circumstances, but also *how he decides* what to do, then we have to know what information *he* is operating with and how it is being processed.

This particular criterion of adequacy (with which we are in complete accord) links our problem directly with a large body of theory and research in cognitive psychology, where an information processing approach to human cognition has been receiving increasing emphasis in recent years. (See Reitman 1965 and Hunt 1962 for detailed summaries of work in this field.) Any theory of natural IP systems which hopes to meet a criterion of cognitive validity will have to take such work into account and be compatible with its established findings. Conversely, what we already know of natural IP systems through ethnographic research must enter into such a theory and its interpretation, and perhaps stimulate some needed reforms in the purely cognitive studies of complex systems.

In brief, the foregoing describes what we have attempted to do in this paper: namely, to propose a formal theory of natural information processing systems of one particular type and to provide it with an interpretation consistent with relevant findings from ethnographic and cognitive research.

The Structure of Information Processing Systems

In the most general sense of the term, an information processing system is a set of interconnected rules of inference, or information processing rules, each of which includes instructions for gathering and operating upon a body of data which becomes the rule's *input information*. Depending upon the nature of the situation being assessed in accordance with these instructions, the input information may take any one of a finite number of possible configurations. The rule also specifies a limited set of potential inferences, each one of which is a possible output of the rule. And finally, it contains a *mapping function* from the various configurations of the input information onto the set of possible outputs.[1]

This characterization of an IP rule covers a wide range of possible types, since we have specified neither the nature and internal structure of input/output information nor the type of operations called for. A rule for obtaining the product of two numbers, for example, would be consistent with this description if it told us how to arrange the input data and how to carry out the procedures necessary for getting an answer. Although such rules would certainly be relevant to a description of how the members of a particular society organize their scientific knowledge, we shall limit ourselves in this paper to a discussion of what we shall call *classification* or *code rules*.[2]

[1] A mapping from a set A *onto* a set B requires a function f such that (1) for every a_i in A there is exactly one b_j in B such that $f(a_i) = b_j$, and (2) for every b_j in B there is *at least* one a_i in A such that $f(a_i) = b_j$. A mapping from A *into* B requires a function such that only condition (1) must hold; e.g., there may be some b_j in B such that for every a_i in $A, f(a_i) \neq b_j$.

[2] During the remainder of this discussion, when we use the terms "information processing system" or "information processing rule," it is to be understood that we are referring to rules or systems of rules of the classification (code) type and not to other kinds of information processing behavior which are outside the scope of this paper.

The inputs and outputs of a code rule shall be referred to as *input situations* and *output situations*, respectively. A situation in general is one member of the set of all possible combinations of values on a finite set of variables, such that each combination includes exactly one value from each variable. A variable of the type used in such a rule (called an *assessment*) describes the set of possible correspondences between a given entity and the member categories of a particular classification scheme. The rule specifies the two sets of assessments which generate the potential input and output situations, and it includes a procedure for determining which of the possible inputs is actually in effect for any given application. It also contains the required mapping from the set of potential inputs onto the possible outputs. The applications of a code rule proceed as follows: The actual input situation is first discovered by following the specified procedure. The mapping then indicates which of the potential output situations should be in effect. The indicated output thus contains information inferred from the existence of a specific input situation.

Even though a classification rule may be used to determine the activity appropriate to a given situation, we should emphasize that the output of the rule is not overt behavior in any form; it is information concerning the classification of one or more entities (i.e., a situation). When we have to use an address term to an unfamiliar Alter, for example, a code rule may be used to determine which category of some classification scheme he corresponds to; the actual use of an appropriate term may be considered an overt behavioral realization of the inferred knowledge. Inferences obtained by applying a code rule of this type may also be put to purely internal use, perhaps in making judgments about the behavior of another individual, or in using a second rule which requires this information to define its input. In other words, a classification rule operates completely at the informational level, even though the inputs and outputs may ultimately be connected to activity of the senses and motor behavior.

Derivation of the Theory

The Structure of Axiomatic Theories In deriving the formal components of this theory, we shall be using the axiomatic method. This approach to theory construction usually involves four basic elements: (1) a specification of primitive notions, (2) a statement of the axioms, (3) a presentation of relevant definitions, and (4) the derivation of useful theorems.

The primitive notions of a theory are those elements that are not defined internally, that is, elements whose nature either is intuitively obvious or can be inferred from other theories taken as logically antecedent to the one being constructed. To state that a certain logical element is primitive does not prohibit us from describing or explicating it, but it does imply that such description or explication is external to the formal theory. Discussion of the primitive notions usually constitutes part of the theory's interpretation: the assignment of "meaning" to otherwise "meaning"-less logical entities.

The axioms of a theory constitute the set of basic propositions from which all other propositions are derived. They must be logically independent of one another and refer to no notions that cannot be reduced to primitive elements. The definitions and theorems form the set of derived elements, and they must be reducible to the axioms and primitive notions.

We have already mentioned the idea of an interpretation for a theory and have indicated that it gives "meaning" to statements contained in the theoretical corpus. It is a true but often overlooked fact that a theory by itself is simply a set of one or more logical statements that fulfill certain formal criteria, but which are "meaning"-less until given an interpretation (see Braithwaite 1962). For example, the statement $a^2 + b^2 = c^2$ may be logically correct as a theorem for some specific theory, but it has absolutely no link with the "real world" until the primitive notions and axioms of that theory are given a representation in terms of particular phenomena. Then, perhaps, we have a relationship true of things labeled "right triangles." Even though a theory and its interpretation are two completely different things, it is often the case that they are presented simultaneously, usually through a judicious choice of familiar terminology. This technique has a tremendous advantage over others when we are dealing with a theory of even moderate complexity; it certainly facilitates understanding on the part of the reader by providing him with ready-made concepts to which he can anchor the derivation. But it also has the disadvantage that ambiguities or contradictions in the interpretation will sometimes be used as an argument for rejecting the theory, when in fact the interpretation alone is at fault and should be corrected or abandoned. This drawback can often be overcome through a careful selection of the primitive notions and axioms. If they are chosen so as to minimize their number and complexity, and to reduce the possibility of ambiguous interpretation, then the derived entities will share this precision and lead to a generally straightforward interpretation of the entire theory. This is one of the principal advantages of a self-conscious, carefully done axiomatic derivation. By constructing an economical axiomatic basis for the theory, we can minimize the amount of interpretation required to make it meaningful, and thereby eliminate many of the semantic problems that might otherwise confront us in using the theory and constructing productive models for it.

Primitive Notions and Axioms[3] The first primitive notion of this theory is that of an *entity*, symbolized E_i. It shall be interpreted very broadly as referring to any phenomenon that possesses a set of properties. An entity specified in a code rule operates as a variable whose current value depends upon the particular circumstances in which the rule is applied. In a detailed example presented in the Appendix to this paper, one of the relevant entities is identified by the label "Alter," or "addressee." "Alter" might refer to different human beings at different points in time, but it must always refer to an individual fulfilling the specified role of addressee.

The second primitive notion, also broadly interpreted, has been labeled *categoriza-*

[3] Most of the notational conventions and formal terminology used here conform with that employed in Suppes (1957).

tion (K_j), and may be regarded as a classification scheme in terms of which some entity or entities may be classified. Any categorization K_p consists of a set of *categories* k_{pq}, each of which refers to one or more properties which an entity may possess. (The first subscript in the notation k_{pq} identifies the categorization as K_p; the second specifies the particular category.) The term "color," for example, might be the label of a specific categorization for native speakers of English, the constituent categories being referred to by the terms "blue," "green," "red," "yellow," etc. *Category* is the third primitive notion of this theory; and, although in the formal sense it may not be completely correct to propose two primitive elements that exhibit such strong interdependence, this state of affairs should present no difficulties in the present context, but will simplify the following discussion immensely.

The fourth primitive notion is that of a *correspondence* between an entity E_i and a category k_{pq}, which we shall represent by the ordered couple $\langle E_i, k_{pq}\rangle$.[4] We say that an entity corresponds to a particular category when it possesses some property or set of properties which allows it to exemplify that category. For example, an entity labelled "social occasion" will correspond to the category labeled "formal" if the attributes or properties of that occasion are such as to make it a "formal social occasion."

The fifth and final primitive notion concerns what we have called the assessment of an entity E_i in terms of a categorization K_p, represented by the ordered couple $\langle E_i, K_p\rangle$. This notion refers to a basic operation in any classification rule, that of determining the actual correspondence between an entity and some category of a relevant categorization. Under this interpretation, a correspondence may be regarded as the result of making an assessment.

The two axioms which constitute the propositional basis of the theory may be stated as follows:

Axiom 1 (Contrast)

Given an assessment $\langle E_i, K_p\rangle$ and categories k_{pq} and k_{pr} in K_p ($k_{pq} \neq k_{pr}$), if E_i corresponds to k_{pq}, then E_i does not correspond to k_{pr}.

Axiom 2 (Finite Membership)

For every categorization K_p there exists some positive integer $n \geqslant 2$ such that K_p has exactly n member categories.

[4] An ordered couple is a special case of an ordered n-tuple: specifically, the case in which there are exactly two elements. In general, an ordered n-tuple is a group of n elements in which *order* as well as *membership* is important. For example, while the set $\{A, B\}$ is the same as the set $\{B, A\}$, it is *not* the case that the ordered couple $\langle A, B\rangle$ is the same as the ordered couple $\langle B, A\rangle$, since two ordered n-tuples can be identical if and only if for every positive integer $i \leqslant n$, the ith element of the first is identical with the ith element of the second. Also, the same element may appear more than once in an ordered n-tuple, while this is not true of a set. For example, we could have an ordered triple $\langle A, B, A\rangle$; the *set* of elements in this triple would be $\{A, B\}$ (or $\{B, A\}$).

Axiom 2 is the simpler of these and can be disposed of quickly. It states that a categorization must have at least two member categories and that it must be finite in extension. If we could not discover a positive integer n, greater than or equal to 2, such that the categorization had exactly n members, then it must have either no members, just one member, or an infinity of members (i.e., no matter how large an n we selected, the set would always have more than n members). (See Suppes 1960: 98) For reasons which should be fairly obvious, we do not want to allow any of these three cases.

Axiom 1 requires that the possible results of making a given assessment must contrast with one another. That is, if one particular correspondence results, then no other correspondence may result simultaneously. Since we interpret an assessment as referring to a process that may be performed at different points in time, and since an entity is a variable that may take any of a limited set of values, we do not have to assume that Axiom 1 requires the result of an assessment to be the same every time the process is repeated. It does require, however, that in any given instance the resulting correspondence must be unique. The axiom therefore places several restrictions on specific models for the theory. We could not allow a case, for example, in which one of the assessments involved a categorization having categories like "formal" and "important" if it were possible for the entity (e.g., a "social occasion") to correspond to both categories simultaneously.

Definitions and Theorems In this section we shall present a series of definitions and theorems which characterizes the structural features of code rules and which has a direct bearing on data analysis, model construction, and testing methods relevant to empirical studies that involve this theory.

Definition 1 (State of an Assessment)
Given an assessment $\langle E_i, K_p\rangle$, an ordered couple $\langle E_i, k_{pq}\rangle$, where k_{pq} is a category in K_p, is a *state* of the assessment $\langle E_i, K_p\rangle$.

Under the interpretation we are proposing for this theory, *a state of an assessment is any possible result of that assessment*; *the actual result in a given instance is a correspondence.* Every state of an assessment is therefore a potential correspondence.

Theorem 1, which follows, is essentially a restatement of the contrast axiom.

Theorem 1 (Contrast between States)
Given an assessment $\langle E_i, K_p\rangle$ and states $\langle E_i, k_{pq}\rangle$ and $\langle E_i, k_{pr}\rangle$ $(k_{pq} \neq k_{pr})$ of this assessment, if $\langle E_i, k_{pq}\rangle$ is a correspondence, then $\langle E_i, k_{pr}\rangle$ is not a correspondence.

PROOF: The proof follows directly from Axiom 1 and Definition 1.

Definition 2 (Assessment Set)
A set A_u of assessments $\langle E_i, K_p\rangle$ is an *assessment set.*

An assessment set, as its name implies, is simply a specified set of assessments.

Definition 3 (State Set)
Given an assessment set A_u, a set S_{uv} of states is a *state set generated by* A_u if and only if for every assessment $\langle E_i, K_p\rangle$ in A_u there is exactly one state $\langle E_i, k_{pq}\rangle$ of $\langle E_i, K_p\rangle$ in S_{uv}.

In other words, by taking exactly one state of each assessment in A_u, and by combining these states into a single set, we obtain a state set generated by A_u.

Definition 4 (Situation)
Given an assessment set A_u and a state set S_{uv} generated by A_u, S_{uv} is a *situation* if and only if for every state $\langle E_i, k_{pq}\rangle$ in S_{uv}, $\langle E_i, k_{pq}\rangle$ is a correspondence.

That is, a situation (in the formal sense that we are using) is a state set, S_{uv}, such that every state in S_{uv} is also a correspondence. Recalling an earlier remark, a state of an assessment is interpreted as a potential result of that assessment, while a correspondence is the actual result. Or, in other words, an assessment is a kind of variable; its states are the values it may take; and a correspondence is its current value. We can extend this idea to the notions of state set and situation by noting that a state set refers to a possible result of making some *set* of assessments, while a situation refers to the actual result. In slightly more formal terminology, a situation is the set of current values taken by the members of a set of variables of the type we have called assessments. The following theorem extends the notion of contrast between states to contrast between state sets.

Theorem 2 (Contrast between State Sets)
Given an assessment set A_u and state sets S_{uv} and S_{uw} generated by A_u ($S_{uv} \neq S_{uw}$), if S_{uv} is a situation, then S_{uw} is not a situation.

PROOF: Since $S_{uv} \neq S_{uw}$, there must be at least one state $\langle E_i, k_{pq}\rangle$ such that $\langle E_i, k_{pq}\rangle$ is a member of S_{uv} and $\langle E_i, k_{pq}\rangle$ is not a member of S_{uw} (Definition 3). S_{uv} is a situation, and by Definition 4, $\langle E_i, k_{pq}\rangle$ must be a correspondence. Since $\langle E_i, k_{pq}\rangle$ is a state of some assessment in A_u and is not a member of S_{uw}, there must be some other state $\langle E_i, k_{pr}\rangle$ of that same assessment which is a member of S_{uw} (Definition 3). Since $\langle E_i, k_{pr}\rangle$ cannot be a correspondence (by Theorem 1), S_{uw} cannot be a situation (Definition 4). Q.E.D.

The following definition introduces the notion of *code segment*, which plays a central role in the derivation and interpretation of the remainder of this theory.

Definition 5 (Code Segment)[5]

Given two finite assessment sets A_i and A_0, the two sets S_i and S_0 of all possible state sets generated by A_i and A_0, respectively, and a many-one function f which maps S_i *onto* S_0, the ordered 5-tuple $T = \langle A_i, A_0, S_i, S_0, f\rangle$ is a *code segment* if and only if the following two conditions are satisfied:

5.1 (MAPPING)

For all state sets S_{ij} in S_i and S_{0k} in S_0, $f(S_{ij}) = S_{0k}$ if and only if when S_{ij} is a situation, then S_{0k} is a situation.

5.2 (MINIMAL DIFFERENCE)

For every assessment $\langle E_p, K_q\rangle$ in A_i, there exists some state $\langle E_p, k_{qr}\rangle$ of $\langle E_p, K_q\rangle$, two state sets S_{ij} and S_{ik} in S_i, and two state sets S_{0u} and S_{0v} in S_0 ($S_{0u} \neq S_{0v}$) such that (i) $S_{ij} \cap \sim (S_{ik}) = \{\langle E_p, k_{qr}\rangle\}$, (ii) $f(S_{ij}) = S_{0u}$, and (iii) $f(S_{ik}) = S_{0v}$.

We shall refer to the two assessment sets A_i and A_0 as the sets of *input assessments* and *output assessments*, respectively; and, similarly, we shall call the members of S_i and S_0 *input state sets* and *output state sets*.

Since a state set may be regarded as a potential situation, it follows that a code segment provides a mapping from one set of potential situations (the members of S_i) onto another set of potential situations (the members of S_0).[6] If a particular input state set is determined to be the actual situation after making the assessments indicated in A_i, then there is a unique output state set that also has the status of a situation. Although it would be possible (from one point of view) to interpret a code segment as one type of classification rule that people might actually carry around in their heads, the psychological evidence indicates that such an interpretation would be incorrect and that a code segment would better be conceptualized as a description of the *capability* of an individual to make inferences within a particular domain.

[5] A note on the selection of terminology might be appropriate before we go any further with this derivation. The term "code" will appear frequently throughout the discussion (as in the labels "code rule" and "code segment"); and, as might be expected, its use follows from several statements made by C. O. Frake (1964a: 132, 133; 1964b: 112), and from the notion of "recoding" discussed by Miller (1956). A "code segment," to illustrate, defines a mapping from one body of information to another, and therefore can be interpreted as describing the way in which elements of the first body are "recoded" (or "encoded") as elements of the second body. If the "cultural code" is to be considered the totality of recoding procedures, then it is apparent that a code segment gives a complete specification of the possible results of using one of these procedures—that is, a "segment" of the "cultural code." As we shall see within the space of a few definitions and theorems, and "ordered code rule" accounts for the mapping expressed in a code segment and gives a rule with which the coding task can be performed.

[6] There is some question as to whether or not it would be better to regard the output of a code segment as a situation or as a single correspondence. In the author's opinion, the latter alternative has the greater intuitive appeal, and thus far suffers from no contrary empirical evidence. Regarding the output as a situation, however, has the greater degree of generality, since we can always specify that A_0 has only one member and that the outputs are therefore potential situations containing only one correspondence apiece. For this reason, S_0 has been defined as a set of potential situations, rather than a set of potential correspondences generated by a single assessment. If this proves to be unsatisfactory, a few simple changes should suffice to correct the problem without altering the structure of the theory in any appreciable manner.

In order to facilitate further discussion of Definition 5, and to provide us with an illustration that we can enlarge upon throughout the remainder of this derivation, the following is offered as an example of a code segment.

EXAMPLE 1 (CODE SEGMENT)
We define a code segment

$$T = \langle A_i, A_0, S_i, S_0, f \rangle,$$

where

$$A_i = \{\langle E_1, K_1 \rangle, \langle E_2, K_2 \rangle\},$$
$$A_0 = \{\langle E_3, K_3 \rangle\},$$

and

$$K_1 = \{k_{11}, k_{12}\},$$
$$K_2 = \{k_{21}, k_{22}\},$$
$$K_3 = \{k_{31}, k_{32}, k_{33}\}.$$

The set S_i of all possible state sets generated by A_i is

$$S_i = \{S_{i1}, S_{i2}, S_{i3}, S_{i4}\},$$

where

$$S_{i1} = \{\langle E_1, k_{11} \rangle, \langle E_2, k_{21} \rangle\},$$
$$S_{i2} = \{\langle E_1, k_{11} \rangle, \langle E_2, k_{22} \rangle\},$$
$$S_{i3} = \{\langle E_1, k_{12} \rangle, \langle E_2, k_{21} \rangle\},$$
$$S_{i4} = \{\langle E_1, k_{12} \rangle, \langle E_2, k_{22} \rangle\}.$$

The set S_0 of all possible output state sets generated by A_0 is

$$S_0 = \{S_{01}, S_{02}, S_{03}\},$$

where

$$S_{01} = \{\langle E_3, k_{31} \rangle\}.$$
$$S_{02} = \{\langle E_3, k_{32} \rangle\},$$
$$S_{03} = \{\langle E_3, k_{33} \rangle\}.$$

The mapping function f is specified as follows:

$$f(S_{i1}) = S_{01},$$
$$f(S_{i2}) = S_{02},$$
$$f(S_{i3}) = S_{01},$$
$$f(S_{i4}) = S_{03}.$$

For example, f indicates that if S_{i1} (the set $\{\langle E_1, k_{11} \rangle, \langle E_2, k_{21} \rangle\}$) is determined to be a situation, then S_{01} (the set $\{\langle E_3, k_{31} \rangle\}$) is the implied output situation.

We can now discuss the operation of the two conditions (5.1 and 5.2) specified in Definition 5 and their relevance for an interpretation of the theory.

Condition 5.1 describes the mapping function f, which relates the two sets of potential situations generated by A_i and A_0. Condition 5.2 performs a less obvious function. It requires for each assessment in A_i that there be two input state sets that differ from one another only in regard to one state of that assessment, and which map onto different outputs. In Example 1, S_{i2} and S_{i4} fulfill this condition for the assessment $\langle E_1, K_1 \rangle$; and S_{i1} and S_{i2} (or S_{i3} and S_{i4}) satisfy the requirement for $\langle E_2, K_2 \rangle$. Suppose, for example, that the definition lacked Condition 5.2. We could then construct a code segment that required an assessment in A_i for which there existed *no* pair of input state sets differing only in the result of this assessment and leading to different outputs. Such an assessment would not differentiate any pair of inputs with regard to their mapping, and hence would be totally irrelevant to the outcome in any application of the associated rule. We could construct an improper code segment of this type by altering K_3 (in Example 1) and S_0, and by changing the mapping function as follows:

$$f(S_{i1}) = S_{01} \qquad (= \{\langle E_3, k_{31} \rangle\}),$$
$$f(S_{i2}) = S_{02} \qquad (= \{\langle E_3, k_{32} \rangle\}),$$
$$f(S_{i3}) = S_{01} \qquad (= \{\langle E_3, k_{31} \rangle\}),$$
$$f(S_{i4}) = S_{02} \qquad (= \{\langle E_3, k_{32} \rangle\}).$$

(K_3 now contains only the members k_{31} and k_{32}.) Note that the result of the assessment $\langle E_1, K_1 \rangle$ is totally irrelevant to the outcome, which is now determined solely by the result of the assessment $\langle E_2, K_2 \rangle$. In summary, Condition 5.2 demands that every assessment called for by a code segment be relevant to the result of the information processing operations. For any interpretation of the theory which involves even a minimal notion of efficiency, a restriction of this type should be required.

Although a code segment has been defined as mapping *complete* descriptions of potential input situations onto potential outputs, it is often the case that full descriptions are not required to account for the mapping, and that if the members of a certain *subset* of states are known to be correspondences, then a unique output is automatically implied. This condition obtains when every potential input situation that contains that subset of states maps onto the same output. In Example 1, we note that this condition is fulfilled for the set of states $\{\langle E_2, k_{21} \rangle\}$, since the only two input state sets that contain this state (specifically, S_{i1} and S_{i3}) map onto the same output, S_{01}. Therefore, the knowledge that $\langle E_2, k_{21} \rangle$ is a correspondence is sufficient to imply that S_{01} is a situation; once this information is obtained, the assessment $\langle E_1, K_1 \rangle$ is irrelevant to the outcome. If such a subset of states is *minimal* (i.e., if it includes no smaller subset which fulfills the above condition), it is called a *simple path* to the particular output involved. The set $\{\langle E_2, k_{21} \rangle\}$, for example, is a simple path to S_{01}. We

Table 1. Simple Paths for Example 1.

Simple Path	To Output	Accounts for State Set(s)
$P_1 = \{\langle E_2, k_{21}\rangle\}$	S_{01}	S_{i1} and S_{i3}
$P_2 = \{\langle E_1, k_{11}\rangle, \langle E_2, k_{22}\rangle\}$	S_{02}	S_{i2}
$P_3 = \{\langle E_1, k_{12}\rangle, \langle E_2, k_{22}\rangle\}$	S_{03}	S_{i4}

say that a simple path *accounts for* the mapping of every state set of which it is a subset. This notion is formally presented in Definition 6.

Definition 6 (Simple Path)[7]
Given a code segment $T = \langle A_i, A_0, S_i, S_0, f\rangle$ and a state set S_{0j} in S_0, a set P_k of states of the assessments in A_i is a *simple path to S_{0j} generated by T* if and only if the following two conditions are satisfied:
6.1
For every S_{iq} in S_i such that $P_k \subseteq S_{iq}, f(S_{iq}) = S_{0j}$
6.2
There is no proper subset P_z of P_k ($P_z \subset P_k$) such that for every S_{iq} in S_i where $P_z \subseteq S_{iq}, f(S_{iq}) = S_{0j}$.

The simple paths generated by the code segment in Example 1, and the state sets and mapping for which they account, are shown in Table 1.

The following theorem shows that for every input state set generated by A_i (in some given code segment T) there is at least one simple path that accounts for that state set.

Theorem 3 (Existence of Simple Paths)
Given a code segment $T = \langle A_i, A_0, S_i, S_0, f\rangle$, and given S_{ij} in S_i and S_{0q} in S_0 such that $f(S_{ij}) = S_{0q}$, there exists a simple path P_k to S_{0q} generated by T such that $P_k \subseteq S_{ij}$.

PROOF: Since the proof is intuitively fairly simple, though rather long in its complete form, we shall present only a sketch of the complete version. Assume that the theorem is false, and that there exists *no* set P_k of states of assessments in A_i such that Conditions 6.1 and 6.2 are fulfilled. Now, S_{ij} cannot be a simple path (by this assumption). Since Condition 6.1 is true for S_{ij}, Condition 6.2 must be false (if true, then S_{ij} would have to be a simple path). Therefore, there must exist some proper subset P_z of S_{ij} such that for every S_{ip} in S_i where $P_z \subseteq S_{ip}, f(S_{ip}) = S_{0q}$. Thus for P_z, Condition 6.1

[7] We can define "subset" and "proper subset" as follows: A set A is a subset of B ($A \subseteq B$) if and only if for every a_i in A, a_i is also a member of B. A set A is a *proper subset* of B ($A \subset B$) if and only if A is a subset of B *and* there is at least one b_j in B such that b_j is not a member of A.

must be true; and since P_x is not a simple path (by assumption), there must be some proper subset P_y of P_x such that Condition 6.1 is true for P_y. The same holds true for any $P_z \subseteq S_{ij}$ such that Condition 6.1 is true. By induction, therefore, we have an infinite series of smaller and smaller proper subsets of S_{ij} which meet Condition 6.1, but which fail Condition 6.2. Since S_{ij} must be finite in extension (Definition 5 and Axiom 2), such an infinite series of proper subsets cannot exist. Therefore, we must reject our initial assumption, and the theorem is proved. Q.E.D.

An important implication of Theorem 3 is that every code segment is capable of generating a set of simple paths which accounts for every potential input situation produced by the input assessments of that segment.

Definition 7 (Simple Code Rule)
Given a code segment $T = \langle A_i, A_0, S_i, S_0, f\rangle$, an ordered 5-tuple

$R = \langle A_i, A_0, P, S_0, g\rangle$,

where P is the set of all simple paths generated by T, and g is a function that maps P onto S_0, is a *simple code rule for T* if and only if the following condition is satisfied:
7.1 (MAPPING)
For every simple path P_k in P, and for every output state set S_{0j} in S_0, $g(P_k) = S_{0j}$ if and only if P_k is a simple path to S_{0j} generated by T.

Since a simple path is interpreted as a set of states minimally sufficient to imply a unique output (every state set that contains the path must go to the same output), we can think of a simple code rule as specifying the absolute minimum information a person would have to know about any given potential input situation in order to achieve the inferential capability described by the associated code segment. We can therefore extend the notion of accountability (mentioned in our discussion of simple paths) to say that if R is a simple code rule for T, then *R accounts for T*. As the next two theorems (4 and 5) will show, if we are given a code segment there is a unique simple code rule which accounts for that segment, and vice versa. It follows that the related problems of deriving a simple rule from a given code segment and deriving a code segment from a given simple rule both have determinate solutions: that is, a solution exists and it is unique. This property establishes the biuniqueness relationship that exists between these two elements of the theory.[8]

Theorem 4 (Existence and Uniqueness of Simple Code Rules)
Given a code segment $T = \langle A_i, A_0, S_q, S_0, f\rangle$, there exists one and only one simple code rule $R = \langle A_i, A_0, P, S_0, g\rangle$ such that R is a simple code rule for T.

[8] Probably the best example of a class of formal structures that have the properties of a simple rule comes from the area of componential analysis. The set of minimal componential definitions resulting from such an analysis—insofar as the analysis produces a true paradigm—and their mapping onto the members of a lexical set, can be expressed in a simple code rule.

PROOF: We shall first prove the existence portion of the theorem (that there is at least one such R), and then the uniqueness portion (that if an R exists, it is the only such R).

The only elements required by R which are not given in T are the set P and the mapping function g. We know that the set of simple paths generated by T exists (Theorem 3). Therefore, we have only to show that g is a function that maps P onto S_0. For this to be true, three conditions must be met:
(i) For every P_j in P, there exists an S_{0p} in S_0 such that $g(P_j) = S_{0p}$.
(ii) For every P_j in P, and S_{0p} and S_{0q} in S_0, [$g(P_j) = S_{0p}$ *and* $g(P_j) = S_{0q}$] implies that $S_{0p} = S_{0q}$.
(iii) For every S_{0p} in S_0, there exists some P_j in P such that $g(P_j) = S_{0p}$.
Condition (i) is satisfied by Definition 7, since any P_j in P must be a simple path to some output. We now move to Condition (ii). Since P_j is a simple path generated by T, there must be some S_{ik} in S_i such that $P_j \subseteq S_{ik}$. (By Definition 5, S_i includes *all* possible combinations of states.) By Condition 7.1, if $g(P_j) = S_{0p}$ and $g(P_j) = S_{0q}$, then P_j must be a simple path to both S_{0p} and S_{0q}. And by Condition 6.1, it must be the case that $f(S_{ik}) = S_{0p}$ and $f(S_{ik}) = S_{0q}$. But according to Definition 5, f is a function, and therefore $S_{0p} = S_{0q}$. Hence, Condition (ii) is satisfied. With regard to Condition (iii), we know that for every S_{0p} in S_0 there must be some S_{ik} in S_i such that $f(S_{ik}) = S_{0p}$ (since f is a mapping *onto* S_0). And by Theorem 3 there must be some simple path P_j to S_{0p} generated by T. Since P is the set of all such simple paths by definition, P_j must be a member of P; and by Condition 7.1, we have $g(P_j) = S_{0p}$. Conditions (i), (ii), and (iii) are therefore satisfied, and there exists a simple code rule R for T.
We now move to the uniqueness problem. By the first part of this proof we know that there is at least one simple rule for T. Assume that R is such a simple rule. Let R' be another simple rule for T. If R is unique, then $R = R'$ and vice versa. Since P must be the same for both R and R' (it is the set of *all* simple paths generated by T), they can differ only in the mapping function. Let

$$R = \langle A_i, A_0, P, S_0, g\rangle$$

and let

$$R' = \langle A_i, A_0, P, S_0, g'\rangle.$$

Since the range and domain of g and g' are the same (P and S_0), they can differ only in the way they map P onto S_0. Now for some P_k in P and S_{0q} in S_0, $g(P_k) = S_{0q}$ if and only if P_k is a simple path to S_{0q} generated by T (Condition 7.1). Similarly, $g'(P_k) = S_{0q}$ if and only if P_k is a simple path to S_{0q} generated by T. Hence, $g(P_k) = S_{0q}$ if and only if $g'(P_k) = S_{0q}$. Therefore, $g = g'$, and it follows that $R = R'$. The simple code rule for T must consequently be unique. Q.E.D.

Theorem 5 (Uniqueness of a Code Segment for a Given Simple Code Rule)
Given a simple code rule $R = \langle A_i, A_0, P, S_0, g \rangle$, there exists one and only one code segment $T = \langle A_i, A_0, S_i, S_0, f \rangle$ such that R is a simple code rule for T.

PROOF: The existence of T follows automatically from Definition 7, since R must be derived from some code segment; and therefore, if R exists, T must exist.

We now move to the uniqueness problem. If there is only one code segment which generates the simple code rule R, then for any pair of code segments T and T' such that R is a simple code rule for both T and T', it must be the case that $T = T'$. This is what we shall attempt to prove. Let R be a simple code rule for some pair of code segments T and T'. Let

$$T = \langle A_i, A_0, S_i, S_0, f \rangle$$

and let

$$T' = \langle A_i, A_0, S_i, S_0, f' \rangle.$$

Since the range and domain of f are the same as those of f', the two functions can differ only in the way they map S_i onto S_0. Now, for every S_{ij} in S_i and S_{0q} in S_0 such that $f(S_{ij}) = S_{0q}$, there must be a P_k in P such that $g(P_k) = S_{0q}$ and $P_k \subseteq S_{ij}$ (Theorem 3 and Definition 7). And by Definition 6 we have $f'(S_{ij}) = S_{0q}$. Similarly, we can show that if $f'(S_{ij}) = S_{0q}$, then $f(S_{ij}) = S_{0q}$. Therefore, we obtain $f(S_{ij}) = S_{0q}$ if and only if $f'(S_{ij}) = S_{0q}$. Hence, $f = f'$, and it follows that $T = T'$. The code segment is therefore unique. Q.E.D.

We now introduce two formal notions that will be required in subsequent definitions.

Definition 8 (State Sequence)[9]
A sequence $Q_i = \langle Q_{i1}, q_{i2}, \ldots, q_{in} \rangle$ of n states $\langle E_p, k_{qr} \rangle = q_{ij}$ is a *state sequence of order n* if and only if for every pair of states q_{ij} and q_{ik} $(j \neq k)$ in Q_i, q_{ij} and q_{ik} are states of different *assessments*.
A state sequence of order n is therefore a sequence of n states of different assessments.

Definition 9 (Initial Subsequence)
Given a sequence $X_i = \langle x_{i1}, x_{i2}, \ldots, x_{im} \rangle$ of m elements, a subsequence $X_i(n) = \langle x_{i1}, x_{i2}, \ldots, x_{in} \rangle$ $(n \leq m)$ of the first n elements of X_i in order is the nth *initial subsequence of* X_i.

For example, if we have a sequence $Z = \langle z_1, z_2, z_3, z_4 \rangle$, the sequence $Z(3) = \langle z_1, z_2, z_3 \rangle$ is the third initial subsequence of Z: that is, the first three elements of Z in order.

[9] For present purposes, we can regard a sequence of n elements as an ordered n-tuple.

Definition 10 introduces the notion of an ordered code rule ("ordered rule," for short). In many respects this is the most important element of the theory, and we shall devote the bulk of the remaining discussion to characterizing its role as the fundamental unit out of which information processing systems are constructed.

Definition 10 (Ordered Code Rule)[10]

Given a code segment $T = \langle A_i, A_0, S_i, S_0, f \rangle$ and the (unique) simple code rule $R = \langle A_i, A_0, P, S_0, g \rangle$ for T, an ordered 5-tuple $R^* = \langle A_i, A_0, P^*, S_0, g^* \rangle$, where P^* is a set of state sequences P_j^* consisting of states of assessments in A_i, and g^* is a mapping from P^* onto S_0, is an *ordered code rule for* R *and* T if and only if the following two conditions are satisfied:

10.1 (ACCOUNTABILITY)

For every P_j^* in P^* and every S_{0q} in S_0, $g^*(P_j^*) = S_{0q}$ if and only if there exists some simple path P_k in P such that $P_k \subseteq s(P_j^*)$ and $g(P_k) = S_{0q}$.

10.2 (ORDERING OF ASSESSMENTS)

For every P_j^* in P^* and for every nth initial subsequence $P_j^*(n)$ of P_j^*, if $P_{jn}^* = \langle E_p, k_{qr} \rangle$, then:

10.2.1

For every P_k^* in P^* such that $P_k^*(n-1) = P_j^*(n-1)$, P_{kn}^* is a state of $\langle E_p, K_q \rangle$,

10.2.2

For each state $\langle E_p, k_{qs} \rangle$ of $\langle E_p, K_q \rangle$, there is a P_k^* in P^* such that $P_k^*(n-1) = P_j^*(n-1)$ and $P_{kn}^* = \langle E_p, k_{qs} \rangle$, and

10.2.3

There exists some P_k^* in P^* and some state sequence Q_k such that $P_k^*(n-1) = P_j^*(n-1) = Q_k(n-1)$; and for all $i > n$, $Q_{k,i-1} = P_{ki}^*$, and for all P_q in P, $P_q \nsubseteq s(Q_k)$.

We shall refer to a state sequence P_k^* in P^* as an *ordered path in* R^* *to* S_{0j} (where S_{0j} is the output upon which the path is mapped), or, more simply, as an *ordered path.*

An ordered code rule differs from a code segment and a simple code rule in that it requires an ordering of assessments. We may interpret this ordering as a temporal one, and as indicating the step-by-step processing of needed information. The choice of each successive assessment depends only upon the results of preceding assessments; that is, upon correspondences already identified up to that point in the process. Suppose, for example, that in applying a hypothetical ordered rule, an individual has already made the assessments $\langle E_1, K_1 \rangle$ and $\langle E_2, K_2 \rangle$ and has determined that the states $\langle E_1, k_{11} \rangle$ and $\langle E_2, k_{23} \rangle$, respectively, are correspondences. At this point he knows that the next correspondence will have to be some state of the assessment $\langle E_4, K_5 \rangle$, for example; and this, consequently, is the next piece of information he processes (see Condition 10.2.2). When he has identified a complete sequence of

[10] We shall use the notation $s(X)$ to indicate the set of elements ordered by some sequence X.

such correspondences (that is, when he has decided which P_k^* in P^* accounts for the actual situation), the mapping indicates the output to be inferred. Such an inference can be made because each complete ordered path must contain all the elements of some simple path (Condition 10.1) which, as we have indicated, contains sufficient information to imply a unique output. It can also be shown, using Condition 10.2, that there must be a "unique beginner": that is, some assessment with which the process begins (see Theorem 6, below).

It should be stressed that while a code segment and its associated simple code rule exhibit the biuniqueness property, this relationship does not necessarily hold between a given code segment (or its simple code rule) and a particular ordered code rule. We can show that there always exists at least one ordered rule for any code segment (Theorem 7), and that for any ordered rule there exists a unique code segment and simple code rule (Theorem 10 and Corollary 10.1). But it can also be demonstrated that the problem of deriving an ordered code rule which accounts for some code segment does not always have a unique solution. In any but the most trivial cases (i.e., where the code segment involves only one input assessment) there is more than one possible ordering of the assessments. The problem of deciding between alternative orderings in empirical studies is an important one and will occupy us later in this paper.

We can exemplify the notion of an ordered code rule and illustrate the possibility of alternative orderings by continuing the development of Example 1. The two possible sets of ordered paths (P^* and $P^{*\prime}$) are given in Table 2, along with their respective mappings. A convenient and easy to read representation of ordered code rules utilizes

Table 2. Two Sets of Ordered Paths for Example 1.

P_j^* in P^* for Ordered Rule R^*	$g^*(P_j^*)$
$P_1^* = \langle\langle E_2, k_{21} \rangle\rangle$	S_{01}
$P_2^* = \langle\langle E_2, k_{22} \rangle, \langle E_1, k_{11} \rangle\rangle$	S_{02}
$P_3^* = \langle\langle E_2, k_{22} \rangle, \langle E_1, k_{12} \rangle\rangle$	S_{03}
P^* in $P^{*\prime}$ for Ordered Rule $R^{*\prime}$	**$g^{*\prime}(P_j^{*\prime})$**
$P_1^{*\prime} = \langle\langle E_1, k_{11} \rangle, \langle E_2, k_{21} \rangle\rangle$	S_{01}
$P_2^{*\prime} = \langle\langle E_1, k_{11} \rangle, \langle E_2, k_{22} \rangle\rangle$	S_{02}
$P_3^{*\prime} = \langle\langle E_1, k_{12} \rangle, \langle E_2, k_{21} \rangle\rangle$	S_{01}
$P_4^{*\prime} = \langle\langle E_1, k_{12} \rangle, \langle E_2, k_{22} \rangle\rangle$	S_{03}

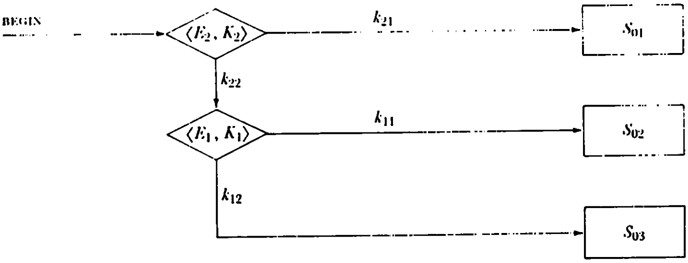

Figure 1. Rule R^*.

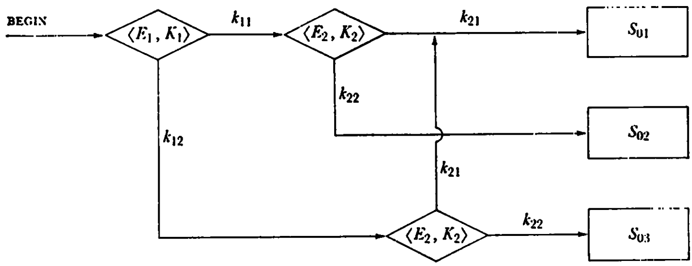

Figure 2. Rule R^{*1}.

some of the notational conventions of computer "flow diagrams."[11] Flow charts for the two ordered rules derived from Example 1 are shown in Figures 1 and 2. The diamond at each node of the diagram represents a single assessment (which is written inside), the labels on the arrows identify the possible correspondences (by indicating the categorization involved), and the rectangular terminating boxes display the potential output situations. Each ordered path is represented in the diagram by a chain of linked arrows originating at the first assessment (the "unique beginner," marked "Begin"), passing through a series of nodes, and terminating at some output. Figure 1, for example, can be read as follows: We start with the assessment marked "Begin" (that is, $\langle E_2, K_2\rangle$). If E_2 corresponds to k_{22} (that is, if $\langle E_2, k_{22}\rangle$ is a correspondence), then the assessment $\langle E_1, K_1\rangle$ is required. If E_1 corresponds to k_{11}, then we have completed the ordered path P_2^* (see Table 2), and the output situation S_{02} is indicated. Other ordered paths would be determined in the same manner.

We can use this notational scheme to simplify our discussion of the formal requirements expressed in Condition 10.2 of Definition 10. To begin with, an nth initial subsequence of some ordered path is represented by a connected chain of arrows originating at the first assessment and following the ordered path through the first n states. (It may, of course, be as long as the ordered path itself.) Suppose that we are

[11] Tree diagrams, which are structurally similar to flow charts, have been used from time to time by decision theorists and cognitive psychologists to represent several different kinds of information processing rules. See Luce and Raiffa (1957), Hunt (1962), and Hunt, Marin, and Stone (1966).

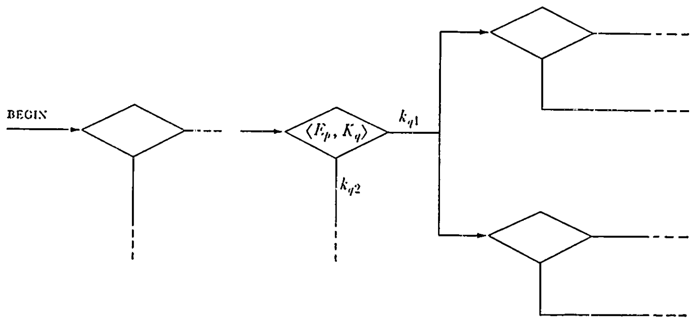

Figure 3. Violation of Condition 10. 2. 1.

given an ordered path P_j^* and some nth initial subsequence of this path, such that the nth arrow of this partial sequence represents a state of the assessment $\langle E_p, K_q\rangle$. The first part of Condition 10.2 requires that every other ordered path which follows the same route through the $(n-1)$th state have some state of $\langle E_p, K_q\rangle$ as its nth element. This must hold true for every nth initial subsequence of every ordered path. In other words, given a partial route through a flow diagram, every continuation of that route must begin with some state of a single given assessment. Condition 10.2 does not permit any ordered rule to have a set of ordered paths which could be partially represented by a diagram like the one in Figure 3. The second part (10.2.2) requires that for every assessment (node) in the diagram, and for each state of that assessment, there must be at least one ordered path that passes through the node and contains that state at the appropriate point in the sequence. If we had an ordered rule that specified the input assessment $\langle E_p, K_q\rangle$, for example, and the states of that assessment were $\langle E_p, k_{q1}\rangle$, $\langle E_p, k_{q2}\rangle$, and $\langle E_p, k_{q3}\rangle$, then no portion of the rule could be represented by a diagram such as the one in Figure 4 (since the state $\langle E_p, k_{q3}\rangle$ does not appear).

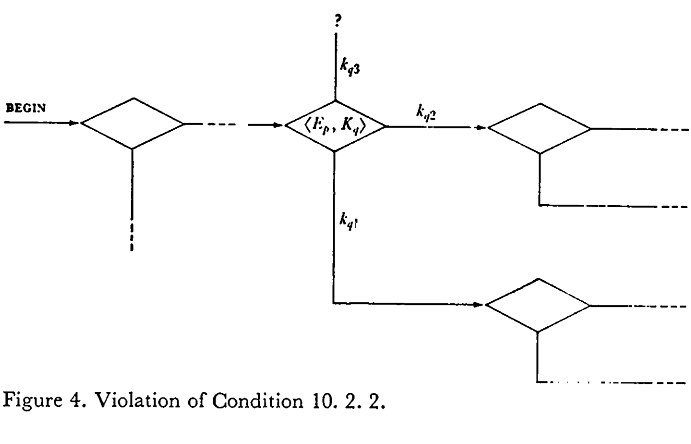

Figure 4. Violation of Condition 10. 2. 2.

The third part of Condition 10.2 demands that every assessment made in accordance with an information processing sequence specified in a given ordered code rule must be relevant to the outcome determined by that sequence. For purposes of illustration, we can temporarily delete this requirement and note the possible consequences. Suppose we are given an ordered path P_j^* in P^* and an nth initial subsequence of this path. We first take the set of all ordered paths that share the route of P_j^* through the $(n-1)$th state (including P_j^* in this set), and delete the nth state from each member. This forms the set of state sequences Q_j mentioned in 10.2.3. If each of these new sequences were to contain all of the states of some simple path given in the simple rule R (hence violating 10.2.3), then clearly the assessment whose states were deleted is irrelevant to deciding between the outputs which could still result following the nth assessment. (Recall that the information contained in a simple path is sufficient to imply a unique output.) The diagram in Figure 5a illustrates a potential ordered rule that violates Condition 10.2.3 at several points (specifically, in the placement of $\langle E_2, K_2\rangle$ in the ordered paths stemming from $\langle E_1, k_{11}\rangle$, and in the placement of $\langle E_3, K_3\rangle$ in the various paths extending from $\langle E_1, k_{12}\rangle$). The simple paths P_j (for the simple rule associated with the possible ordered rules incorrectly represented in Figure 5a) are as follows:

$P_1 = \{\langle E_1, k_{11}\rangle, \langle E_3, k_{31}\rangle\}$ (to S_{01}),
$P_2 = \{\langle E_1, k_{11}\rangle, \langle E_3, k_{32}\rangle\}$ (to S_{02}),
$P_3 = \{\langle E_1, k_{12}\rangle, \langle E_2, k_{21}\rangle\}$ (to S_{03}),
$P_4 = \{\langle E_1, k_{12}\rangle, \langle E_2, k_{22}\rangle\}$ (to S_{04}).

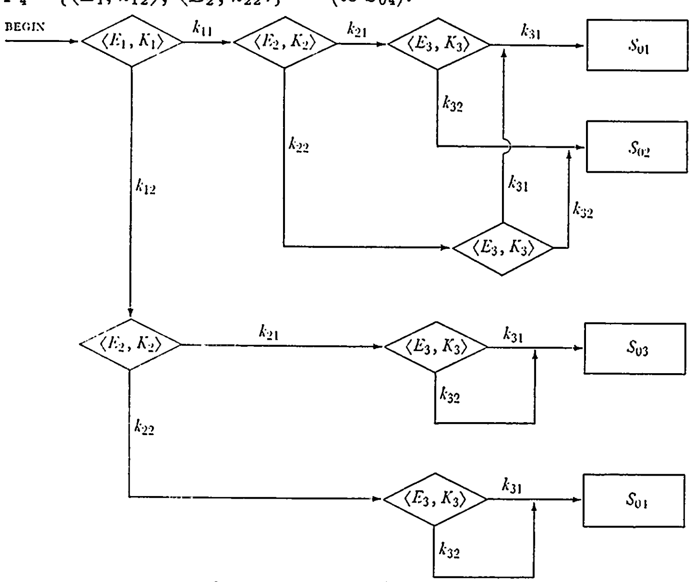

Figure 5a. Violation of Condition 10. 2. 3.

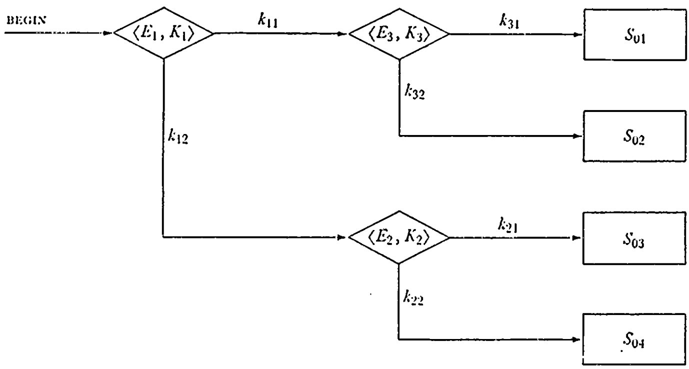

Figure 5b. Corrected to adhere to Condition 10. 2. 3.

If in all ordered paths starting with $\langle E_1, k_{11}\rangle$ in Figure 5a the states of $\langle E_2, K_2\rangle$ are deleted, each of the resulting sequences contains the states of either P_1 or P_2. Although this assessment is relevant in all ordered paths starting with $\langle E_1, k_{12}\rangle$, it is not relevant once the correspondence $\langle E_1, k_{11}\rangle$ has been determined. A more obvious violation of Condition 10.2.3 concerns the placement of $\langle E_3, K_3\rangle$ in the paths stemming from $\langle E_1, k_{12}\rangle$. This assessment contributes no information toward making a decision between S_{03} and S_{04}; this can be seen in the diagram or by studying the simple paths listed above. An ordered rule in which these violations are corrected is shown in Figure 5b.

We can now introduce several theorems relevant to the structure and existence of ordered code rules.

Theorem 6 (Unique Beginner)
Given an ordered code rule $R^* = \langle A_i, A_o, P^*, S_o, g^*\rangle$, there is an assessment $\langle E_p, K_q\rangle$ in A_i (the unique beginner) such that for every ordered path P_j^* in P^*, P_{j1}^* is some state of $\langle E_p, K_q\rangle$.

PROOF: The proof follows directly from Condition 10.2.1. For every pair of ordered paths P_j^* and P_k^* in P^*, $P_j^*(0) = P_k^*(0)$ (since this is the empty sequence). There must be some assessment $\langle E_p, K_q\rangle$ such that P_{j1}^* is one of its states. If P_{j1}^* is some state of $\langle E_p, K_q\rangle$, then by Condition 10.2.1, P_{k1}^* is also a state of this assessment. Since we have shown that this is true of any pair of ordered paths in P^*, it is true of all ordered paths in P^*. Q.E.D.

Theorem 7 (Existence of an Ordered Code Rule)
Given a code segment T and the (unique) simple code rule R for T, there exists at least one ordered code rule R^* such that R^* is an ordered code rule for R and T.

PROOF: To demonstrate the existence of such an ordered code rule, we need only show that it is possible to construct a set of ordered paths using the assessments in A_i, and to provide a mapping from this set to the members of S_0, such that the conditions of Definition 10 are satisfied. We shall sketch such a construction procedure here. First of all, we know that an ordered rule is derived from a given code segment and its associated simple code rule. To begin the construction, take all of the input state sets (the potential input situations) of the code segment and order their members in the same way, such that the first state of each sequence is a state of some assessment $\langle E_i, K_p \rangle$, the second state of every sequence is a state of another assessment $\langle E_j, K_q \rangle$, and so on. Map each of these sequences onto the output to which its corresponding input state set is mapped by the function f. Figure 5a, to which we referred earlier, represents such a set of sequences and shows that they produce a complete "tree" diagram or flow chart. Each sequence contains the elements of a simple path (Theorem 3) and hence fulfills Condition 10.1. They also fulfill Conditions 10.2.1 and 10.2.2 (which are needed to produce a complete tree of the type shown in Figure 5a). The set of sequences, however, may not adhere to Condition 10.2.3 (as we illustrated earlier). If this is the case, then irrelevant assessments are dropped from each path in which they contribute no relevant information. Such assessments must be deleted one at a time, and the modified structure should be checked for additional violations after each deletion. Since the set of potential ordered paths changes after each incorrect assessment is dropped, the mapping will have to be revised according to Condition 10.1. It should be clear, however, that this requirement (10.1) and the first two parts of 10.2 will still be fulfilled. When no additional deletions are required, then all of the conditions of Definition 10 will be met, and an ordered code rule for the given code segment will have been constructed. Figure 5b illustrates the result when this procedure is applied to the set of sequences shown in Figure 5a.

We now move on to the proof of several theorems that illustrate the accountability of a particular ordered code rule for its associated code segment and simple code rule.

Theorem 8 (Accountability of Ordered Paths for Situations)
Given a code segment $T = \langle A_i, A_0, S_i, S_0, f \rangle$, the (unique) simple code rule $R = \langle A_i, A_0, P, S_0, g \rangle$ for T, and $R^* = \langle A_i, A_0, P^*, S_0, g^* \rangle$ an ordered code rule for T and R: for every P_k^* in P^* and for every S_{ij} in S_i such that $s(P_k^*) \subseteq S_{ij}$, and for every S_{0q} in S_0, $g^*(P_k^*) = S_{0q}$ if and only if $f(S_{ij}) = S_{0q}$.

PROOF: If $g^*(P_k^*) = S_{0q}$, then by Condition 10.1 there must be some P_j in P such that $P_j \subseteq s(P_k^*)$ and $g(P_j) = S_{0q}$. Now if $s(P_k^*) \subseteq S_{ij}$, then $P_j \subseteq S_{ij}$. Since P_j is a simple path to S_{0q} generated by T (Condition 7.1) and $P_j \subseteq S_{ij}$, it must be true that $f(S_{ij}) = S_{0q}$ (Condition 6.1). Similarly, if P_k^* is an ordered path, there must be some P_u in P such that $P_u \subseteq s(P_k^*)$ (Condition 10.1). If $s(P_k^*) \subseteq S_{ij}$, then $P_u \subseteq S_{ij}$. Let $g(P_u) = S_{0z}$. By Condition 6.1, $f(S_{ij}) = S_{0z}$. But since we are given that $f(S_{ij}) = S_{0q}$, and since f is a function (Definition 5), we have $S_{0z} = S_{0q}$. Hence $g(P_u) = S_{0q}$ and $g^*(P_k^*) = S_{0q}$ (by Condition 10.1). Therefore, $g^*(P_k^*) = S_{0q}$ if and only if $f(S_{ij}) = S_{0q}$. Q.E.D.

Theorem 9 (Existence and Uniqueness of Ordered Paths)
Given a code segment $T = \langle A_i, A_0, S_i, S_0, f \rangle$ and an ordered code rule $R^* = \langle A_i, A_0, P^*, S_0, g^* \rangle$ for T: for every state set S_{ij} in S_i there exists one and only one ordered path P_k^* in P^* such that $s(P_k^*) \subseteq S_{ij}$.

PROOF: Since the proof of this theorem is long and overly tedious, it is not presented here. The proof is, however, very similar to the one used in Theorem 3; and the interested reader is referred there for a sketch of the procedure.

Theorem 8 shows that an ordered path accounts for the mapping of any potential situation of which its elements form a subset. Theorem 9 demonstrates that for any potential situation S_{ij} defined by a code segment T there exists exactly one ordered path P_k^*, specified by a given ordered code rule for T, such that the elements of P_k^* form a subset of S_{ij}. These two theorems jointly imply that all potential situations defined by some code segment T can be accounted for by an ordered code rule for T. Theorem 10 (following) shows that for any ordered code rule R^* there is one and only one code segment T such that R^* is an ordered code rule for T; and hence that the problem of determining the code segment accounted for by an ordered rule has a unique solution.

Theorem 10 (Uniqueness of a Code Segment for an Ordered Code Rule)
Given an ordered code rule $R^* = \langle A_i, A_0, P^*, S_0, g^* \rangle$, there is one and only one code segment $T = \langle A_i, A_0, S_i, S_0, f \rangle$ such that R^* is an ordered code rule for T.

PROOF: The proof is identical in form to the one presented for Theorem 5 and is not given here. The reader is referred to the earlier proof for an illustration of the procedure.

Corollary 10.1
Given an ordered code rule R^*, there is one and only one simple code rule R such that R^* is an ordered code rule for R.

PROOF: The proof follows directly from Theorems 4 and 10.

Although there exist a unique code segment and a unique simple rule for any given ordered rule (the subject of the last several theorems), the converse is not generally true. For any given code segment or simple code rule there will usually be more than one permissible ordered code rule. (Figures 1 and 2 illustrate a case in point.) Consequently, when we attempt to apply this theory to the description of a natural system, we are presented with the problem of choosing between alternative and equally valid (at least in the formal sense) analyses. We might want to ask, therefore, whether or not there exists an interesting subclass of ordered rules which has a greater likelihood of being represented in natural systems. In this connection we can enlarge upon an idea about human cognition that we alluded to earlier in the paper, and discuss its effects in generating such a subclass of ordered rules. The assumption involved can be stated as follows: *Human beings tend to process information in such a way as to minimize the long-run average number of items processed.* If we can interpret the processing of an item of information to be equivalent to making a single assessment, then the implication of this assumption is clear: individuals will tend to modify the internal structure of their ordered rules in such a way as to minimize the average number of assessments performed. This idea has been used so far to justify the "relevancy" condition in Definition 5 (Condition 5.2), which says that no code segment requires an assessment irrelevant to all decisions between outputs; and it can also be used in justifying Condition 10.2.3 in Definition 10, which states that at any stage of an information processing sequence the next assessment made must be relevant to the possible outcomes at that stage.

From the above assumption we can also derive the following proposition: *At any stage in an information processing sequence, the next assessment to be made minimizes the average number of subsequent assessments which must be made before an output can be determined.*[12] In terms of the tree diagram or flow chart representation of an ordered rule, this statement says in effect that the assessment at any given node is chosen so as to minimize the number of assessments in the branches that emanate from that point. We say that any ordered code rule that meets this requirement is an *efficiently ordered rule.* Without going into detail, an efficiently ordered rule can always be constructed for a given simple code rule by operating upon the set of simple paths contained in the latter. Construction proceeds in a step-by-step fashion by first selecting a unique beginner (the first assessment) and then, for each branch coming from this node, selecting a second assessment, and so on, such that each choice conforms to the efficiency requirement. The process is terminated in each branch as soon as the ordered set of states contains all the elements of a simple path and, therefore, is sufficient to indicate a single output.

Returning to Example 1 and the set of simple paths shown in Table 1, we note that if the assessment $\langle E_2, K_2\rangle$ is chosen as the unique beginner, then one of its states ($(\langle E_2, k_{21}\rangle)$) completes the simple path P_1 and leads directly to the output S_{01}. The other branch from this node must lead to the assessment $\langle E_1, K_1\rangle$ before a simple path is accounted for. This produces the ordered rule shown in Figure 1. If, on the other hand, the assessment $\langle E_1, K_1\rangle$ were taken as the unique beginner, then *both* branches which

[12] This proposition actually involves an additional simplifying assumption which states, in essence, that the potential input situations in a code segment have an equal "probability" of occurrence. In empirical cases this is obviously untrue, but the errors of analysis that it is capable of producing are relatively minor and occur only in very specific and limited circumstances. Since the potential errors are minimal, the fact that this assumption makes possible the detailed analysis of empirical cases is sufficient justification for its use.

emanate from this node must lead to $\langle E_2, K_2\rangle$ before a simple path can be completed. The ordered rule that this generates is shown in Figure 2. The first process (producing rule R^*) obeys the efficiency requirement by picking a unique beginner that minimizes the number of subsequent assessments (only one additional assessment in one branch), while the second process does not (it requires an additional assessment in each of *two* branches). The former thus generates an efficiently ordered rule, as an inspection of Figures 1 and 2 should demonstrate. The algorithm that this procedure illustrates can be applied to simple rules of any degree of complexity to derive an efficiently ordered rule. In some cases more than one solution is possible, but it should be stressed that the range of alternative analyses is greatly reduced, usually to a set of ordered rules that differ from one another only in the relative sequencing of two adjacent assessments.

The notion of an efficiently ordered rule is an important one with regard to the theory's potential application in ethnographic description. If the efficiency assumption is justified—and available evidence indicates that it is—then we should expect to find that information processing routines actually in use by given individuals can be described in a valid manner by efficiently ordered rules. Since a simple code rule can be determined through elementary frame elicitation techniques, and since efficient rules can be derived in turn from simple rules, then it follows that we should have in hand a technique that requires data that are relatively easy to come by, and which produces descriptions showing a reasonable approximation to cognitive validity. Several tests performed in conjunction with the example presented in the Appendix to this paper indicate, though on a limited basis, that the approximation falls within quite acceptable limits.[13]

One other point deserves mention before we conclude this part of the discussion. It concerns a phenomenon we have called "recoding," following the usage of Miller (1956). In essence, recoding refers to information contained in the output of one ordered rule being used as part of the input information to a second rule. Since a given output is a set of states (a potential situation), and since the input of a rule is also a set of states, then the possibility of recoding is permitted by the formal structure of this theory. For example, in applying one ordered rule, there may be a required assessment for which the necessary information is not immediately available (i.e., it is not known which state of the assessment is a correspondence). If there exists a second rule that contains the states of this assessment in its output (if the assessment is a member of A_0), it can be employed to determine the actual correspondence. In such a case, we say that the input information to the second rule has been recoded in terms of a

[13] These tests consisted of sorting tasks in which the subject was asked to group terms on the basis of their similarity in use. After an initial partition was formed, the subject was asked whether or not any of the groups could be further partitioned into smaller subgroups, and then if any of the initial groups could be placed together in larger groupings. Since a set of ordered rules also generates partitions of a hierarchial type, the test could be used as evidence for the cognitive validity of the set of derived rules.

given output. Because of certain inherent limitations on the amount of information that an individual can process at any one time, some sort of recoding must take place in an information processing system of more than minimal complexity. This capability must also extend to any theory of human information processing which is to be used in the production of valid models for natural systems.

A Note on Cognitive Aspects Although this paper has been primarily concerned with a theory useful in the production of certain types of ethnographic statements, and is by no means a self-contained treatise on the psychology of thinking, we have already seen that our attempt to provide an interpretation for its formal notions depends heavily on a basic commitment to certain ideas about how people think and about their capabilities and limitations in organizing and processing specific kinds of information. This commitment involves such fundamental notions as the itemization of information (i.e., the cognitive representation of information as discrete units and not as continuously variable magnitudes), the sequential processing of information, the tendency toward efficient cognitive systems, limitations on the amount of information that can be processed at one time, recoding, contrast between the states of an assessment (Axiom 1), and so on. In several cases, these ideas have had to be modified or generalized to conform with the implications of well-founded ethnographic theory.

Unfortunately, limitations of space and the fact that this subject is outside the somewhat limited scope of the paper force us to postpone until a later time any detailed discussion of the relationship between this theory and cognitive psychology. We can say, however, and without any great reservations, that we have tried to keep the theory and its interpretation as thoroughly consistent as possible with relevant areas of cognitive and ethnographic theory. For the most part, the attempt has been successful; though there have certainly been cases in which simultaneous compatibility could not be maintained. These cases are interesting in themselves, since they ultimately reduce to conflicts between ethnography and cognitive psychology; but, again, they fall outside the scope of this paper and would be more appropriate to a general discussion devoted to the relationship between these two domains of study. We can only say that a discussion of this type is needed and, hopefully, will not be long in coming.

Appendix. Bisayan Terms of Personal Address

The set of efficiently ordered rules which appears in this Appendix is included solely for the purpose of exemplifying the theory developed in this paper. This example is not intended as a complete and/or valid ethnographic statement concerning the terms of personal address used by any group of Bisayan speakers.[14] It should be stressed that

[14] Bisayan (or Visayan) is a Philippine language in wide use among Christian Filipinos on Mindanao and the central group of islands known as the Visayas. Local dialects can vary widely within this area. The terminology analyzed here is most representative of the islands of Cebu and Leyte and their immediate environs.

the terminology and usage reported here is derived from interviews with a single informant under nonfield conditions, and should not be considered representative of any "typical" speaker of the informant's dialect, nor, perhaps, of the informant himself under more natural conditions.

Even with these reservations, however, the description is sufficiently adequate to indicate something of the complexity of naturally occurring information processing systems and their potential range of variation. In addition, there is some evidence (from the sorting task described in footnote 13) that the description does approximate a valid model of one portion of the informant's "cognitive map," and to this extent can be considered adequate.

The notation used in the following table and diagrams differs slightly from that presented in the body of the paper. Categorizations have been represented by capital letters which have as much mnemonic value as possible, and their member categories are symbolized by the same letters in lower case with identifying subscripts.

The entities and categorizations used in these rules are listed below in Table A1. The various assessments employed are shown in Figures A1 through A6.

Table A1. Entities and Categorizations for the Bisayan Example.

	Entities	Gloss
	E_1	"Alter," "addressee"
	E_2	"social occasion"
	E_3	"proname" (name or name substitute)
	E_4	"first name" (of Alter)
	E_5	"nickname" ("pet name") (of Alter)
	E_6	"social relationship with Alter"
	E_7	"personal background of Alter" (linguistic, "accultuative")
	E_8	"language" (appropriate)
Categorizations	**Categories**	**Gloss**
A		"relative age"
	a_1	"younger"
	a_2	"same"
	a_3	"older"
A'		"absolute age"
	a'_1	"young child"
	a'_2	other (contains several categories not differentiated in these rules)

Table A1 *continued*

Categorizations	Categories	Gloss
C		"membership in Ego's group of 'friends'"
	c_1	"member"
	c_2	"not a member"
E		(types of formal pronames)
	e_1	(Impossible to gloss; represents part of the proname
	e_2	classification scheme. Refer to Rule R_e^*.)
	e_3	
	e_4	
F		"permission of familiarity"
	f_1	"familiarity permitted"
	f_2	"familiarity not permitted"
K		"knowledge" (of some entity)
	k_1	"known"
	k_2	"unknown"
L		(Classification by language)
	l_1	"English"
	l_2	"Bisayan"
	l_3	"Spanish"
M		"marital status"
	m_1	"married"
	m_2	"unmarried," "single"
P		(Personal address classification scheme. Possible
	$p_1 - p_{19}$	correspondences are indicated in Rule R_p^*, P_1 through P_{19}.)
Q		"request for intimacy" (involves a standard verbal formula)
	q_1	"request made"
	q_2	"request not made"
S		"relative status"
	s_1	"lower"
	s_2	"same"
	s_3	"higher"

Table A1 *continued*

Categorizations	Categories	Gloss
T		"possession of academic/professional title"
	t_1	"has title"
	t_2	"does not have title"
U		"degree of intimacy" (with Ego)
	u_1	"intimate friend"
	u_2	"casual acquaintance," "stranger"
V		"types of informal pronames"
	v_1	(See discussion of categorization E.)
	v_2	
	v_3	
W		"absolute social status," "wealth"
	w_1	"middle class," "poor"
	w_2	"high class," "wealthy"
X		"sex"
	x_1	"male"
	x_2	"female"

Note: The terms *ʔángga ʔ*, "nickname," "pet name," *pangalan*, "first name," and *titolo*, "academic/professional title," indicated in Rule R_p^* are not address terms *per se*, but refer to forms which will vary with the particular Alter involved.

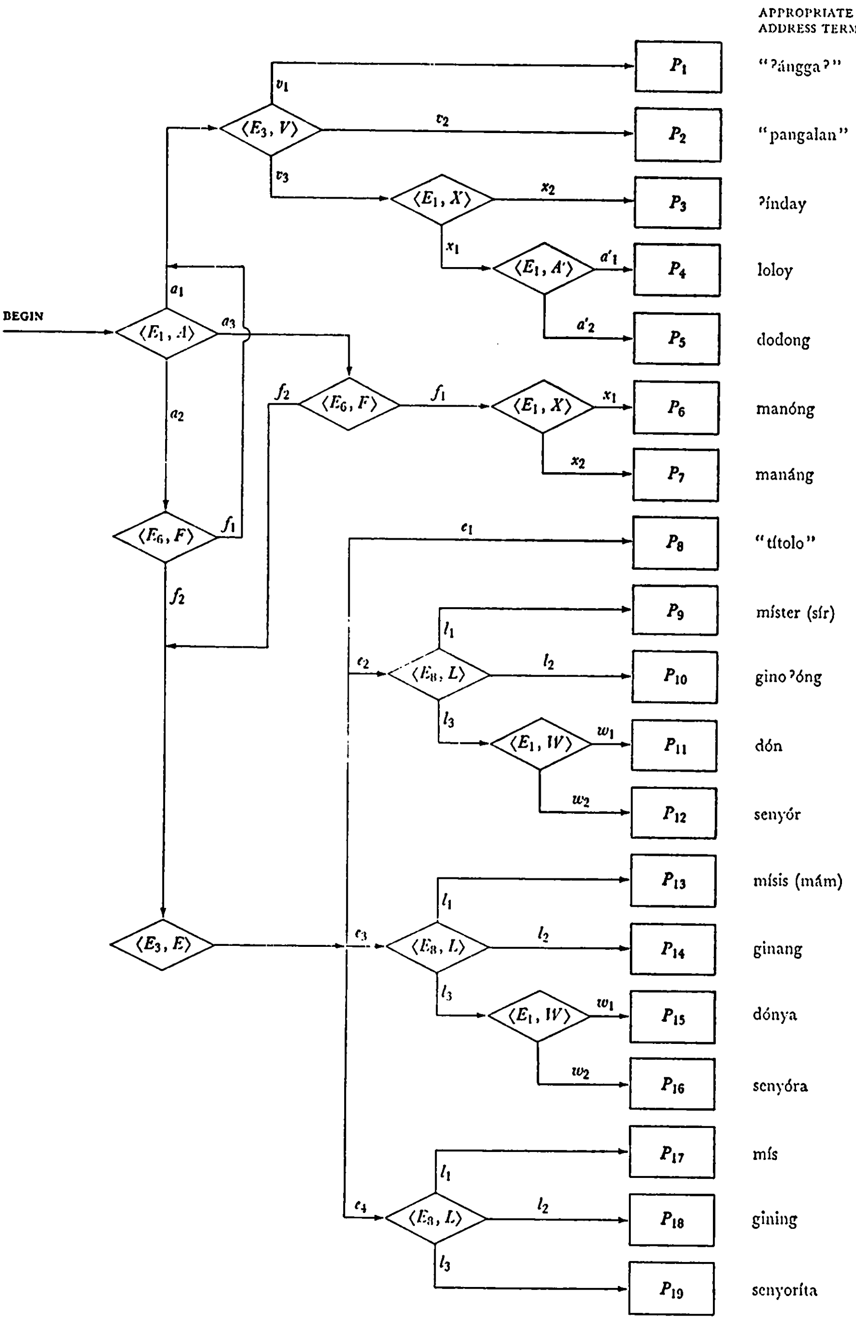

Figure A1. Rule R_p^* for making assessment $\langle E_1, P\rangle$.

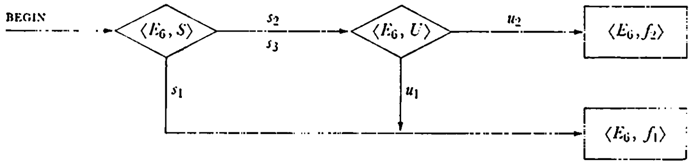

Figure A2. Rule R_f^* for making assessment $\langle E_6, F\rangle$.

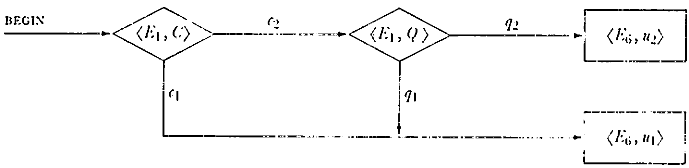

Figure A3. Rule R_u^* for making assessment $\langle E_6, U\rangle$.

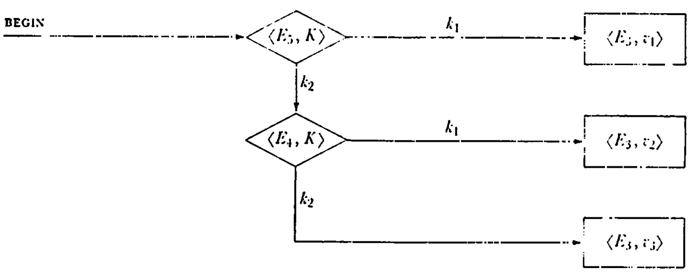

Figure A4. Rule R_v^* for making assessment $\langle E_3, V\rangle$.

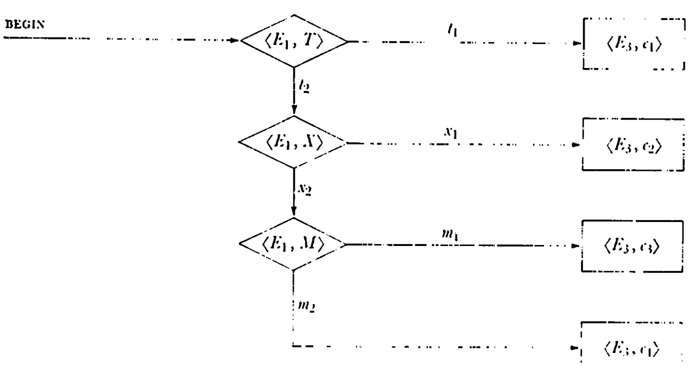

Figure A5. Rule R_e^* for making assessment $\langle E_3, E\rangle$.

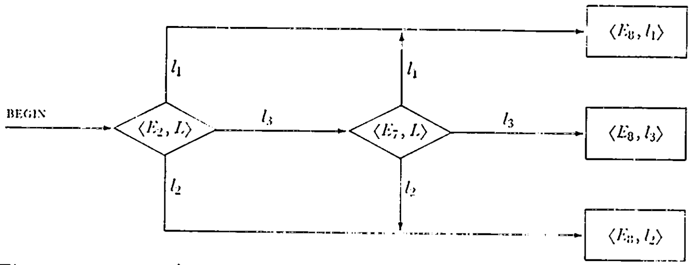

Figure A6. Rule R_i^* for making assessment $\langle E_8, L\rangle$.

References

Braithwaite, R. B.
1962 "Models in the Empirical Sciences," *in Logic, Methodology and Philosophy of Science*, Ernest Nagel, Patrick Suppes, and Alfred Tarski, eds (Stanford, Cal.: Stanford University Press), pp. 224–231.

Frake, Charles O.
1964a "Notes on Queries in Ethnography," *American Anthropologist*, 66, No. 3, Pt. 2: 132–145.
1964b "A Structural Description of Subanun 'religious behavior,'" in *Explorations in Cultural Anthropology*, Ward H. Goodenough, ed. (New York: McGraw-Hill Book Company), pp. 111–129.

Hunt, Earl B.
1962 *Concept Learning, an Information Processing Problem* (New York: John Wiley & Sons).

Hunt, Earl B., Janet Marin, and Philip J. Stone
1966 *Experiments in Induction* (New York: Academic Press).

Luce, R. Duncan, and Howard Raiffa
1957 *Games and Decisions: Introduction and Critical Survey* (New York: John Wiley & Sons).

Miller, George A.
1956 "The Magical Number Seven, Plus or Minus Two: Some Limits on our Capacity for Processing Information," *Psychological Review* 63 : 81–97.

Reitman, Walter R.
1965 *Cognition and Thought, an Information Processing Approach* (New York: John Wiley & Sons).

Suppes, Patrick
1957 *Introduction to Logic* (Princeton, N.J.: D. Van Nostrand Company).
1960 *Axiomatic Set Theory* (Princeton, N.J.: D. Van Nostrand Company).