AN INFORMATION PROCESSING THEORY OF VERBAL LEARNING

Edward A. Feigenbaum

The RAND Corporation
Mathematics Division

P-1817

9 October 1959

PREFACE

EPAM is a theory of human verbal learning, expressed in a language for a digital computer, IPL-V. EPAM was first conceived by Dr. Allen Newell and Dr. Herbert A. Simon, during their early thinking about computer simulation of cognitive processes. They have both helped me considerably in the formulation and implementation of the present EPAM model.

My work on EPAM at Carnegie Institute of Technology was supported by a Ford Foundation Fellowship. Support for the development of IPL-V and the realization of EPAM on an IPL-V Computer has been provided by The RAND Corporation. Concerning the application of IPLs to complex programming problems, the many suggestions of Mr. J. C. Shaw and Mr. Fred Tonge of RAND have been most helpful to me.

I wish to thank the various people who have helped to stimulate my thinking about this research at various stages, especially Dr. H. J. Leavitt, Dr. J. G. March, and my colleagues of the IPL-V Users Group, Julian Feldman and Robert Lindsay.

The indefatigable Miss Jessie Gutteridge typed this manuscript, always with a smile.

Of course, the expression of my appreciation to all of these people does not transfer to them the responsibility for errors in this paper, which I keep wholly for myself.

E. F.

SUMMARY

This paper presents a theory of some elementary forms of human symbolic learning -- memorization, discrimination, association, and attention direction. This theory is concerned with mental activity at the level of the processing of information symbols, which are the basic units manipulated.

The precise statement of the theory is given in the language of a digital computer, specifically as a set of programs in IPL-V*, called EPAM.**

The paper deals generally with information structures and processes for discrimination and association learning, and specifically with behavior in the standard rote learning task. A number of implications of the theory in rote learning situations are explored; comparisons are drawn between the behavior of human subjects in these situations and the behavior of the EPAM model.

---

* Information Processing Language

** Elementary Perceiver and Memorizer

CONTENTS

INTRODUCTION

This paper reports a theoretical study of human verbal learning. The theory herein developed is framed in terms of the functioning of the human mind as an information processing device. It employs a recently developed method of specifying hypotheses about human cognition as programs for a digital computer. Consequences of a set of hypotheses are observed when a system of such programs is made to behave in the presence of "environmental" information.

Thus, first and foremost, we have been motivated by a desire to understand more about the information processing involved in human learning: how the human being obtains information about external objects; organizes this information in a memory; associates information symbols together; and produces responses to stimuli. As a working guide, we have adopted a principle of limited information-processing capacity. The capacity of the human mind to process information is very small compared with the superabundance of potential stimulus information in an environment and a normal human memory.

A second purpose of this study concerns the problem of constructing an intelligent machine as a set of programs for a digital computer. This problem has been the object of much research recently. Problem solving programs (e.g. Newell, Shaw, Simon: 1956, 1959; Gelernter and Rochester, 1958) and perception programs (Selfridge and Dineen, 1955; Rosenblatt, 1959) have been developed, but not much work

has been done on a theory of the acquisition of new information symbols from an environment, the building of discriminations among stimuli, and the storage of information symbols by association. Our research may possibly contribute to the study of intelligent machines in two ways:

a) by suggesting that because the human mind seems to be an effective device for acquiring and storing knowledge about environments, perhaps a fruitful way of proceeding initially on the problem of building a machine which will do these kinds of things is to study and simulate the elementary information processes of the mind.

b) by proposing information processing mechanisms which, in themselves (i.e. without reference to human learning theory), may be useful in the development of intelligent machines.

Third, we hope that the learning model herein presented will be of some interest to students of human decision-making. Empirical decision theorists are fundamentally concerned with how decision alternatives are acquired and how stimuli evoke alternatives. The EPAM model developed in this paper gains, organizes, and associates stimulus and response alternatives; it is a simple decision-making system. We suggest that part or all of EPAM* have some potential as a model of simple human decision-making.

Fourth, we have been motivated to experiment with computer

simulation as a methodology for exploring and understanding elementary human cognition. This method has been used to build a theory of high-level human problem solving (Newell, Shaw, Simon, 1958) and of simple human forecasting (Feldman, 1959), but has not before been used to study elementary human learning. In this paper, we will present the "raw material" for the reader to form his own judgement of the value of computer simulation as a method for building theories of human cognition.

Finally, for the reader who is interested in the real details of this simulation, we hope that the inclusion of the actual programs for EPAM in IPL-V will be of some value. EPAM is the first model to be programmed in IPL-V, the only IPL* which is a "public" computer language. Therefore, it makes sense to complete the presentation of the model by including the actual programs for the use of those who either wish to dig deeply into this model, or wish to understand more about IPL-V for their own research purposes.

The paper is organized in the following way:

Part 1 presents some of the general structures and processes of our information processing model, out of which a theory of verbal learning is later built. The general framework has been abstracted from the human learning setting on the presumption that by itself it

may be of interest to computer scientists, automata theorists, and decision-making theorists.

In Part 2, we apply the framework to a theory of human verbal learning in rote learning tasks. The emphasis here, much more than in Part 1, is on the psychology of human learning.

Part 3 presents the IPL-V programs for the EPAM model. A section of descriptive material is included to facilitate the understanding of the computer programs.

Part 4 develops a number of consequences of our information processing model in standard rote serial learning tasks. Also given are the results of laboratory experiments with human beings in the same tasks. Again, the emphasis is primarily on the psychology of human learning.

The conclusion briefly indicates the direction in which further research on this project will continue.

PART 1

ELEMENTARY ADAPTIVE SELECTION PROCESSES

1.0 INTRODUCTION

The purpose of Part 1 is to develop a theoretical framework for the analysis of human verbal learning in terms of information processes. A model for an elementary type of decision-making will be presented and in Part 2 will be used in a theory of human verbal learning. The model has been abstracted from the verbal learning setting and presented independently in this Part because we feel that it has some interest in its own right as an information processing system, and some generality which would be obscured if the model were presented directly in the context of verbal learning.

1.1 SIMPLE CHOICE PROCESSES

Recently, decision-making models of considerable complexity have been developed for making decisions in problem areas rather narrowly defined (Newell, Shaw, Simon: 1956, 57, 58, 59; Tonge, 1958; Rochester and Gelernter, 1958). One characteristic of all of these models is that they are information processing systems which organize and select information symbols.

The decision-making processes used by these systems generally involve complicated methods of problem analysis and often millions of elementary information processing operations. Some of these systems have been used to simulate human problem solving activity in high-order intellectual tasks (Newell, Shaw, Simon: 1959).

In this Part we examine an information-processing model for a decision-making process much simpler than these. We may label this the simple choice process, and the model involved an elementary selection model. To attempt to define precisely how elementary is an elementary selection, or the simplicity of a simple choice process is a difficult task, for the criteria will vary somewhat from example to example. Our purpose will be better served if we leave the simple choice process as undefined for the moment and provide, instead, some examples of so-called simple choices.

1.1.1 Examples of "simple choices"

1.1.1.1 Computer Programs

In almost all computer programs written, one can find instances where the program is required to take one of various courses of action depending on the states of certain information storage registers in the computer. Machine instruction codes and computer languages generally provide one or more conditional transfer commands for this purpose. The selection of the appropriate course of action is usually accomplished by executing a series of conditional transfers. In this process, one course of action--the subcode to which the program eventually transfers--is selected.

1.1.1.2 Recognition of Letters of the Alphabet

A subject sits before a projection screen. Letters of the alphabet are flashed upon the screen. As each

is flashed up, he announces which letter it is. He is perceiving certain characteristics of the flashed letter, and selecting from a set of alternatives (the alphabet he knows), the particular alternative which fits the characteristics he has perceived.

1.1.1.3 Selection of Rules to Apply, By General Problem Solver I

The General Problem Solver (Newell, Shaw, and Simon; 1959) solves problems in symbolic logic by applying certain methods to subproblems it generates from the problem it is given. The methods involve the application of rules of transformation to logic expressions. In a given situation, GPS is faced with the problem of selecting, for application, one rule out of a set of rules given it. GPS analyzes a particular state of affairs and a desired state of affairs, distills from this a set of characteristics for a rule which will bring about the desired state from the given state, and selects a rule having these general characteristics.

1.2 SOME FEATURES OF THE SIMPLE CHOICE PROCESS

We turn now to an abstract consideration of some of the features of a mechanism that makes simple choices. Such a mechanism will, of course, be operating in some environment, and for our purposes we will characterize the environment strictly by what information it presents to the mechanism.

In general, the environment will be rich with information. The choice mechanism has a finite and limited capacity to process information; therefore, it will generally be true that the environment will contain a great deal more information than the mechanism is capable of considering at the time of choice.

1.2.1 Encoding

The superabundance of stimulus information in the environment burdens the information processor of limited capacity with the problem of reducing the amount of information to manageable proportions. This process of information reduction we shall call encoding. In encoding, the mechanism essentially asks certain questions of a stimulating informational pattern--questions corresponding to dimensions along which it hopes to classify the stimulating pattern. The string of "answers" to these questions represents the code for the pattern. Suppose a processer were interested only in categorizing the serial pattern of strings of symbols and used the following codes:

no alternation	1
single alternation	2
double alternation	3

It would encode the sequence: pppppppppppppppppppp as 1; the sequence: qrqrqrqrqrqrqr as 2; the sequence: ssttssttssttsstt as 3.

1.2.2 Alternatives

A simple choice mechanism must have a set of

alternatives from which to choose. How it acquires these alternatives is a separate question in itself, and a model for this will be discussed later. However, the set of alternatives which it has available can generally be regarded as the product of its experience with stimulus environments in the past.

The subject recognizing letters has had to learn the alphabet previously. The computer program which branches many ways at a given point has either been given specifically the branching alternatives, or it has had to generate these itself previously, so that at branch time the set of branching alternatives is precisely specified.

1.2.3 An Organization of Alternatives

Given a set of stimulus codes (1.2.1) and a set of alternatives (1.2.2), the mechanism must also have some system of organizing its alternatives in such a way that it can map coded stimulus information onto subsets of its alternatives. A simple model of such an organization will be presented subsequently.

1.2.4 Further Information Constraints

If the environment presents a profusion of information patterns, it may be necessary to introduce a further constraint on the information intake, which we shall call the "focus of attention." The focus of attention picks out of the environment those patterns which the encoding process will code. This feature of behavior is so common

to human beings that it hardly needs to be elaborated at the level of examples.

Another constraining process is one which we might call the "code-noticing" process. Just as the focus of attention dictates what external information the encoding process will scan, so the code-noticing process determines which portions of the internal stimulus code will be considered by the internal process.

To summarize, the general ingredients of a simple choice process are these: external to the process, an informational environment, generally rich with potentially stimulating information; a focus of attention for allocating the effort of an encoding process, a process which reduces external stimuli to internal codes; a code noticing filter which passes certain portions of internal codes to a mapping process which uses a system of organization to map stimuli (internally coded) onto subsets of a set of choice alternatives which the mechanism possesses.

1.3 PROCESS MODEL OF ELEMENTARY SELECTION

In this section, we develop a model for the selection process discussed above.

1.3.1 Alternatives, Codes, Tests

We consider a system of information symbols organized into a set of alternatives, A_i.

As previously discussed, environmental stimuli are

encoded by an encoding process. The result is a set of internal codes, or just "codes." Let C be an element of this set. The code C will be assumed, for purposes of this model, to be a binary code.

A test shall be defined as a process which examines a characteristic of a code; its output can have n values, corresponding to the paths of an n-way branch. In this particular model, we shall deal only with tests whose output is 0 or 1 (+ or -, etc.); i.e., for which n = 2.

An Example of a Binary Tree of Tests and Alternatives

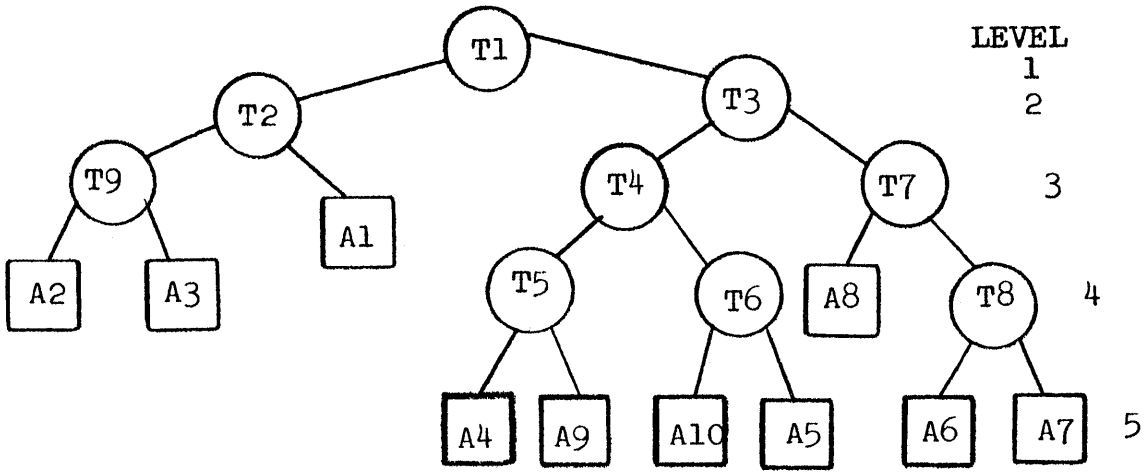


FIGURE 1.1

1.3.2 Organization of Alternatives and Tests

Tests and alternatives are organized into an entity which we shall call a decoding net, or a selection net. A decoding net is a tree of tests and alternatives. The size of the tree is measured by the maximum level to which the tree is elaborated. Any

path through the tree terminates in an alternative, and an alternative can occur only at the termination of a path. All other objects along the path are tests.

An example of a binary tree of tests and alternatives is shown in Figure 1.1.

1.3.3 Selection Process

The selection process is an iteration over a selection net. It selects a path through the net by activating, in sequence, a string of tests of a code C (the argument). The activation of a test provides the selection process with a value for a branch (for a binary net this is + or -, branch left or branch right). Taking the appropriate branch results in the acquisition of the next test to be activated, and so on, until the path terminates in an alternative. This alternative is the selected alternative.

If A is the space of all alternatives in the selection net, it will be seen that the action of each successive test is to reduce, in its turn, the subspace of A selected by the previous tests. In the binary tree shown, T_1 acts to select either the subset $A^{(1)} \equiv \left\{ A_1, A_2, A_3 \right\}$ or the subset $A^{(2)} = \left\{ A_4, A_5, \ldots, A_{10} \right\}$. T_2 selects from $A^{(1)}$ either $A^{(3)} = \left\{ A_1 \right\}$ or $A^{(4)} = \left\{ A_2, A_3 \right\}$, and so on. Obviously, only certain partitions of A into subsets are allowed by such a system of selection. The alternatives at levels greater than r along a particular branch of the

tree can be characterized by the enumeration of the tests at the levels 1 through (r-1), and the value of the (r-1) level test.

Suppose that the selection process keeps a record of the values of successive tests which it activates. For any selected alternative, A_j, this record is a unique pattern representing the position of A_j in the tree. Within a particular tree of alternatives, this record is a kind of "diminished code" for A_j in this tree. If there were no redundancy in the code C, then the "diminished code" would, of course, contain the same number of bits as C.

1.4 ORGANIZATION FOR SELECTION

We have realized the selection model presented in the previous section on a digital computer, a device capable of performing the selection process we have just specified. The computer realization will be described in detail in Parts 2 and 3. Briefly, into the memory of a computer is loaded a set of alternatives (either "active" alternatives, i.e. symbols which are the names of programs, or "passive" alternatives, i.e. symbols which are pure symbolic entities). Also loaded is the set of tests T, which are computer programs. The selection net organization is imposed on tests and alternatives, and a program, the selection process, works over the tree, using codes as input, and producing selected alternatives as output.

The computer realization of our model has the great advantage of being specific, and thus forms an excellent base upon which to build a discussion of the features of the selection system described. For the remainder of the paper, we shall identify the formal model with the computer realization and treat these, for purposes of our discussions, as identical.

1.4.1 Selection by Position vs Selection by Association

Both William James and John von Neumann investigated the organization of information in a memory, though for two different "systems" having a memory. James, in Principles of Psychology, attempted to discover the organization of human memory. von Neumann's task was to design an organization of the memory of a sequential, stored program digital computer. (Goldstine and von Neumann, 1947, 1948) Fundamentally, both investigations related to the same problem: how to organize the information which a behaving system has acquired so that retrieval of this information at some future time can be achieved efficiently.

The answers given were quite different. James' model of human memory was framed strictly in terms of associations among stored information. Retrieval of any particular piece of information stored in the memory could be accomplished only by the detective-like process of tracing through associated information, a procedure which is familiar to everyone. Von Neumann's

model for the memory of a stored-program computer turned out to be much different from the Jamesian model, for he founded his system, and thus virtually all modern day computing systems, on the well-ordering properties of the positive integers. In von Neumann's system, memory was divided into n cells, which were explicitly identified with the positive integers 1 through n. All access to information stored in cells was accomplished by reference to the address of a cell; i.e., the integer assigned to the cell. Retrieval of information thus became connected with the integral position of an item of information in a series.

Examples of access by position proliferate in modern computer technology. Arrangements for access by association are quite rare.

Examples:

1. In regional addressing schemes, such as that used in the IT compiler language, variables are uniquely identified by their relative position in a defined region. Thus the variable Y5 is precisely the sixth cell in the Y region; i.e., the cell whose address is obtained by adding five to the defined address of Y0. (Perlis, et. al., 1957)

2. In virtually all matrix computation schemes, the elements of matrices are stored in linear succession in storage registers of the computer, either row-wise or column-wise. To retrieve the $(i, j)^{th}$ element of the

matrix, one performs a computation using the dimensions of the matrix to arrive at the address of the number.

3. Some computer systems have attempted to get away from the heavy dependence on the integer properties of storage cells. Thus all list languages, like IPL-V, (Newell, et. al, 1959), automatically tie storage cells into lists, and make no use of storage cell addresses as sequential integers. In one sense, then, symbols stored together on IPL lists constitute information stored by association. It is very frequently the case, however, that a symbol is retrieved from a list not on the basis of some characteristics of the symbol, but by its integer position on the list. Thus the order code of IPL-V contains a set of instructions: J8n ... Find the n th symbol on list (0).

4. The IPL languages do, however, contain one pure example of information storage by association, and this constitutes one of the few such examples that one can cite. IPL lists have associated with them entities called description lists. Description lists contain symbols (called values) which are classified entirely on the basis of a characteristic which they possess (these characteristics are called attributes). The commands which retrieve information from description lists require only a specification of the lists and the characteristic. The position of the symbol on the

description is irrelevant.

We feel that the difference between memory storage arranged positionally and memory storage arranged by association is fundamental. Experience has demonstrated that the former arrangement limits extremely the flexibility of behavior of complex programs. The larger computer memories get, and the more involved memory address housekeeping becomes, the more burdensome will become such selection systems.

The model formulated above is an attempt to examine the behavior of a completely associational elementary selection system. Its most important property is that it provides for selection of symbols based entirely on the properties of information stimulating the selection. It is the antithesis of the address register hardware of a present-day computer. We shall see later what kinds of flexibility are available to such a system for expanding its set of evokable alternatives over time on the basis of its experience.

Suppose we wish to build a selection model as a means of exploring the selection processes which the human brain uses to produce simple responses to informational stimuli. We find it highly implausible to assume that such a model of the human brain can be built of addressable cells and arithmetically computable links.

There is no evidence from brain physiology to suggest that anything like this exists. Furthermore, experience with digital computers suggests that operating with such "hardware" requires very detailed pre-planning of memory allocation, greatly limits flexibility of storage and retrieval, and is inefficient for the processing of large amounts of information.

We feel that an addressing scheme which is completely non-direct, such as the one which characterizes the model we have presented, will prove to be much more fruitful in building models of human retrieval and discrimination processes. In Part 2, we will present one such model, and in Part 4 examine whether or not this hunch is borne out.

1.4.2 Cue Codes and Associated Alternatives

So far, the term "association" has been used to characterize the process whereby a stimulus (code) is mapped by tests onto an alternative (or set of alternatives). We will now extend the concept slightly to include associations between the alternatives in the space A themselves. In other words, we seek a process which, using virtually the same selection machinery, will select a sequence of alternatives, which alternatives we will then call "associated."

The extension is as follows:

We shall store with an alternative, A, at its

place in the tree, a set of codes. In this set will be a code corresponding to each context in which the mechanism is capable of being instructed to associate alternatives. (Consider a human example: one can instruct a subject to "free associate" on DOG in the context of ANIMALS). Each code in the set (we shall call it a cue code) is a partial copy of the code which is capable of evoking some other alternative by the usual selection procedure (some of the redundancy has been removed). When the system is stimulated to produce a string of associated alternatives in a context Y, it selects the first alternative (using the stimulus and the selection process we have already discussed). When it has found this, it chooses from the set of cue codes stored with the alternative, that cue code corresponding to the context Y. This cue code is then used as the stimulus code for a second selection, producing an associated alternative, and so on. The process terminates when the "next" cue code for the context Y cannot be found.

Thus, with this simple extension, the elementary selection mechanism can be made to produce not only "immediate" associations, but also "removed" associations generated as chains of associations.

1.5 LEARNING PROCESSES

At the heart of the selection system we have described

is the selection tree by means of which a set of alternatives is organized. In all of our discussions so far, we have been considering only non-changing trees; i.e., we have described for our system no processes which would allow it to grow its selection tree and thereby expand its set of alternatives. In this section, we will propose some modifications and additions to the basic system which will allow it to do just this.

What we will have, then, is a system that is capable of self-modification of its behavior. From experience with an informational environment exhibiting considerable variety, it will expand its repertory of responses it can make to stimuli. When a system behaves in this way, we commonly refer to its behavior as learning. Though such labeling is convenient and justified, it is also somewhat dangerous, for it may coax us to ignore the possibility that so-called learning may really be just a catch-all word for many different things. We will not be caught in this trap, and will in fact discuss a number of different kinds of learning (or alternatively a number of different phases of the overall learning) of which our system is capable.

1.5.1 Extensions of the Model

1.5.1.1 Code Images

We will now allow that with each alternative, A_i, stored in the selection tree there is stored also an image of the code C, capable of "selecting" A_i.

This image may be the complete code itself, or it may consist of only part of the code, or it may be some internal translation of the code into another code, etc.

1.5.1.2 Matching Process

We postulate the existence of a matching process which will compare in detail two codes (or a code and a code image) and produce as output a difference between these codes, if one of more differences exist. For example, if the codes to be compared each consisted of a set of characteristics having binary values, then the matching process might report that there is a difference between two codes on characteristic "alpha."

1.5.1.3 Branch Growing

A necessary adjunct to a process which finds differences between codes is a process which creates tests to capitalize on these differences, thereby adding to the "discriminability" of the net. We provide, then, a branch growing process. This process accepts as input a difference between two codes, creates the appropriate test on this difference, and adds this test to the net in a way which we will presently describe.

1.5.2 Expanding the Set of Alternatives

Though the space of alternatives has, in some sense,

been identified with the set of terminals of branches in the selection tree, not much else has been said about the nature of these alternatives. Suppose you were building a net to handle the selection of alpha-numeric characters used by some computing machines. You would certainly provide separate branches for the letters of the alphabet and the digits 0 through 9. Perhaps you might also provide for some special characters. If you are wise, you will probably also create some "empty" branches to cover the possibility that at some future time someone may wish to add other characters to the set with a minimum of effort. In other words, the alternatives in your net will consist of "real" alternatives and "null" alternatives capable of being converted into real alternatives at some later time. Suppose we instruct the system to learn that alternative A is associated with stimulus code C. It may already have learned this, but it must check. The check is performed, obviously, by selecting the alternative currently associated with the code C. There can be only three results of this check. First, the alternative selected may be A, and there is nothing more to do. Second, the alternative selected may be a null alternative. As we have seen, what this means is that a branch has already been provided at a previous time for a new alternative. The system has only to store, in place of this null alternative, the alternative A, which is now

capable of being evoked by C. Third, and most interesting, is the case where the alternative selected by C is non-null and different from A. Call this alternative B. A confusion is imminent for the system unless it takes corrective action, for A and B are different alternatives associated with different stimulus codes. The potentiality for a confusion arose because, although the two stimulus codes may have contained enough information to separate them, there were not enough tests in the net (or, alternatively, the tests that were there did not scan the right characteristics) to catch the distinction.

The action necessary at this point is supplied by the new machinery we have added. The code image at B is compared with the code C by the process which produces a difference between them. If no difference can be found, then a confusion <u>will</u> take place, and there is nothing that can be done about it. It is a fundamental error which this system will make.

If a difference can be found, the branch growing process will then use this difference to create a test which will take advantage of this difference to provide two different sub-branches for A and B. If B was at the r^{th} level of the net, the branch growing process will insert this test in place of B at the r^{th} level, and store A and B at the $(r+1)^{st}$ level, along their correct branches. The net has been grown to preclude future confusions between

A and B. If we wish to provide "empty" branches against the possibility of some future confusions in this area of the tree, we could have the matching process produce more than one difference, and have the branch growing process create and insert a number of tests, those after the first being redundant, and therefore having associated with each of them an "empty" branch.

Having described a system which will expand its set of selection alternatives by self-modification of its "classification" apparatus, we can give another interpretation to this process. The selection tree, at any given time, represents a system of the "diminished codes" which we mentioned earlier. As the number of alternatives increases, codes will have to be lengthened or expanded to encompass the larger set. Reflect on the binary number system. The integers 0, 1, 2, 3, are classified by two bits, the integers 4, 5, 6, 7, need three bits apiece, 8, 9...15 four bits, etc. If more alternatives are to be added to the set which our system recognizes, then the system must have a way of "adding bits" to its classification system. This is what the matching and branch growing processes are doing. If we start our system initially with no tree at all, and instruct it to learn n alternatives (i.e. build a selection tree by the above described processes), and if these

alternatives each contain a large amount of information, the result will be a set of reasonably efficient (but not optimal) diminished codes for the n alternatives. Our selection mechanism with its self-modification processes is a recoding device.

1.5.3 Four Kinds of Learning--Familiar Names Revisited

Into our original system, we have introduced a whole subsystem of dynamics. The larger system has certain self-organizing and modifying features that enable it to adapt to changing environmental stimulating conditions by enlarging its repertory of alternatives with which it can respond. We have referred to such behavior as learning, defined broadly. We can do better than this. Unlike some learning models which have been proposed in the past, the specificity of the one presented in this paper allows one to state, with some precision, different kinds of "learning" of which the system is capable. Let us take the tour.

1.5.3.1 Memorization Learning

Certainly the simplest learning process a system can exhibit is that of gaining and holding new information. This is purely a storage process. One almost hesitates to call this learning, because one can easily think of systems which do this, systems of which one would be reluctant to say, "They learn." Yet if this so-called "memorization"

results in changed behavior on the part of the system doing the memorization, it is justifiably called learning. The end purpose of all the dynamic machinery which we have proposed is to facilitate this process of memorization. The system goes through the processing of differencing and branch growing solely for the purpose of creating a unique terminal point of the net for storing a new alternative.

1.5.3.2 Discrimination Learning

We have seen in the previous section that a major problem faced by the adaptive processes of the system is the resolution of internal confusions arising from the nature of the selection tree and the decoding process. When two alternatives become confused, the usual reason for this is that not enough information has been scanned to distinguish between the two. This situation is properly called a "failure to discriminate." The corrective action taken by the system--the growing of the tree at the terminal of confusion--results in the system learning (permanently) the difference between the two confused alternatives. The system learns to discriminate among its growing set of alternatives. Memorization learning, as we have seen, consists of the addition of alternatives to the net (where the place of the addition has already been provided); discrimination

learning, analogously, consists of the addition of tests to the net along branches subject to confusion.

1.5.3.3 Association Learning

In the section on associations (1.4.2) we described a variation of the basic mechanism which would allow for the linking of two or more alternatives of the system in an "association." For an alternative y to be associated with alternative x, some or all of the code capable of evoking y was stored with x. When x is evoked, this code can be obtained and used as the argument of the selection process, thereby evoking y. If x is considered "stimulus information" and y is considered "response information," then we have a convenient and simple way of describing the process by which internal responses are "hooked up" to external and internal stimuli. This is just the association process described. The process of obtaining and storing coded "response" information (about y) with a "stimulus" alternative (x), we are calling association learning. It is a second-order learning process in that it must make use of information and net structure already established by the memorization and discrimination learning processes, but it is nevertheless important because it adds flexibility of behavior to the system, particularly in building relatively complex "output" sequences from simple elements.

1.5.3.4 Learning to Learn

The three kinds of learning discussed above involve modification or expansion of the objects and operators of the system. The one part of the system which we have not yet subjected to processes of change is the set of information processes themselves. Dynamic modification of the information processes--the selection process, the matching process, the branch growing process--constitutes a fourth kind of learning. We have not mentioned this earlier, for essentially it constitutes a diversion from the main point of this paper. Yet, it might be of some interest to note just a few of the ways in which the information processes can be modified to the benefit of system performance.

1.5.3.4.1 Noticing

A code is presented and decoded. When the decoding is ambiguous (i.e. a confusion ensues), a matching process examines the confused code and code image for differences. But the matching process, like all others, is a serial process, and must look for differences by scanning the codes in some order. We shall call this the noticing order. Certainly learning efficiency will be increased if the matching process notices first those portions of codes

which are most likely to contain differences. But such clues cannot, in general, be known a priori. An efficient noticing order can be obtained only after the system has had some experience with stimuli. Even this is not enough, for the qualitative nature of the stimuli may change over time, so that what was previously an efficient noticing order is no longer efficient. Obviously what we need to do is to specify a process by which the system itself may order its noticing priorities on the basis of its experience with stimuli.

One such scheme is as follows. The noticing order is identical to a list-ordered set of general code elements (the first element in the list in noticed first, the second element second, and so on). When an element is noticed and a difference found there, this element gets "promoted" up one position on the list. Even if we initialize the system with an arbitrary noticing order, after a while those elements which are most promising in terms of differences existing at these elements (and this "promise" is heavily dependent on recent past history) will get scanned before those elements which are less promising.

This reordering of the noticing order is, in fact, an important kind of learning on the information processes, because it allows for efficient scanning of codes which are generally elaborate and highly redundant, and for adaptation so that efficient scanning can continue to take place as the stimulus environment changes.

1.5.3.4.2 Time-sharing

The serial information processor of limited capacity for processing information is faced with a fundamental problem. The environment is constantly presenting it more information than it needs, faster than it can possibly handle this information. We have already considered ways in which the system can regulate the amount of information it takes in and uses. But this is only one part of the problem. The other part concerns the allocation of processing time to the subprocesses in the system, most of which are potentially insatiable information processers, if set loose without effort-limiting or effort-directing controls.

We shall use the word "heuristic" to designate these control rules. For though they are not rules or methods specifically for limiting

search, they are nevertheless "rules of thumb" for guiding behavior, rules which are successful most of the time for producing efficient behavior, but which sometimes fail, creating trouble for the system.

One set of such information processing heuristics concerns the assignment of the total processing time available in a particular interval among the various stimuli upon which the system is asked to select or learn. Thus, if the system is required to add a number of new stimuli being presented to its repertory, it must decide in which order to process these; whether to process them one at a time to completion, or process each partially, processing the set a number of times; and so on.

Perhaps a more important set of information processing heuristics are the effort-limiting heuristics. These are heuristics for deciding when a particular subprocess has completed its task "satisfactorily." Simon (1957) has discussed the need for "satisficing" rules in behavior systems of limited computational capacity. Several problem-solving computer programs make fundamental use of the concept of satisfactory levels of performance as a

heuristic for limiting effort (see, for example, the Newell, Shaw, Simon Chess Player I (1958) and Tonge's Heuristic Line Balancer (1958)).

Satisficing heuristics play an important role in the adaptive selection process described above. For example, one such heuristic must decide how much information decoding capacity is to be added to the selection net when a branch is grown. When two alternatives are being associated via the net, an effort-limiting heuristic must decide when enough cue information has been stored so that the association really exists.*

As still another example, when an alternative is being memorized in the net, not all the information available about the alternative need be stored (because of the generally high redundancy) but only enough information so that the alternative can sometime later be evoked without ambiguity.

---

* It is clear that these rules provide no guarantee as to the ultimate success of their mission. The heuristic which limits cue information may decide on a satisfactory set of information at time t, but at time t + n this information may be insufficient because of learning which has taken place in the intervening n time periods.

The "cut-off points" intrinsic to these effort-limiting heuristics can be made amenable to modification. One can define processes which will change these cut-off points on the basis of the ongoing experience of the system. For example, suppose the system is learning associations between alternatives, and that we test it periodically and give it information about how well it is doing. The following situations are conceivable. The system may decide that it is making too many errors, or doing too poorly by some other measure, in which case it might take corrective action by raising its cut-off level with respect to the amount of cue information stored to link two alternatives. On the other hand, it may decide that it is doing too well; i.e., the fact that it is making too few errors indicates that it might be able to "get away with" less processing. In such a case, it may revise downward the aforementioned cut-off level. In both cases, performance, let us say, measured in total learning time for the task, may be improved. In short, satisficing heuristics are important to an adaptive mechanism, and the revision (based on experience) of the "cut-off points" constitutes

a major part of so-called "learning to learn," the modification of the information processes in the direction of more efficient processing procedures.

1.5.3.4.3. Additional Cues

In a previous section, we considered the order in which the system notices information and how this relates to efficient processing. The assumption was that the information to be noticed existed in coded form internally. Of some importance to the system is a process for determining what parts of the stimulus are acquired in coded form. Another way to say this is that some process is needed to determine just precisely what the stimulus is. If a letter of the alphabet is presented to the system, is the stimulus merely the letter itself (so that just its characteristics will be coded), or is the stimulus the letter plus a variety of other "cues" and additional information available at presentation time? Such a process becomes particularly important when stimuli are presented which are identical in certain characteristics and can be differentiated only with reference to the local context of cues in which each is presented. A stimulus

may evoke one alternative in some context of cues and another in a different context.

It is clear that there must be some process whose job it is to keep the system aware of local contextual information and that such a process will contribute substantially to the efficient performance of the system. As an illustration of one such process, let us assume that a number of "dimensions of contextual information" have been defined for the system. These may be ordered on a list. When a confusion arises which cannot be resolved by the coded information which is "ordinarily" used, a signal may be given this context-scanning process to provide additional cue information, in the priority order of its list of contextual dimensions. If any of these dimensions provides information useful in the resolution of the difficulty, this dimension gets promoted on the list of dimensions, in a manner precisely analogous to the way in which the noticing order was modified. In such a way, the system can adapt for the efficient use of additional contextual information.

1.6 SUMMARY

A model of an elementary selection process was presented. The elements of the model are: a set of alternatives; a set of tests; codes, which are the internal coded representations of external stimuli. Tests and alternatives are organized in a tree structure, called a selection net, in which the alternatives are found at the terminals of branches of the tree. A selection process decodes the code in the tree, mapping codes onto alternatives (or sets of alternatives). Any such mapping defines an association between a code (coded stimulus) and an alternative.

A scheme by which the alternatives themselves can be associated via the selection net was presented. It was stated that the selection net represents a model of a storage system in which information retrieval takes place by association and not by position, as has generally been the case in computer memory systems.

The model was expanded to include a set of processes for enlarging the set of alternatives. This was conceptualized in terms of "branch growing" in the tree to provide additional storage positions for the new alternatives. In connection with these adaptive processes, four particular types of learning in this system were discussed. These are: memorization learning; i.e., the storage of new alternatives, where place has already been provided for such storage; discrimination learning ,the process by which potential confusions between

old and new alternatives are resolved, and storage places provided for new alternatives, thereby creating discriminations among similar alternatives; association learning, or the building of informational links between alternatives by means of the mechanism of the selection process; and learning-to-learn (self-modification of the information processing to improve the efficiency of performance), discussed in terms of noticing order for examining codes, context-scanning for acquiring additional cues, and heuristics (effort-directing and effort-limiting) for information processing.

PART 2

AN INFORMATION PROCESSING MODEL FOR HUMAN VERBAL LEARNING

2.0 INTRODUCTION

In Part 1 of this paper, we developed a decision-making system capable of associating environmental stimuli with alternatives; expanding its set of alternatives; and of associating one alternative with a set of others. Further, we discussed four kinds of learning of which this system is capable: memorization learning; discrimination learning; association learning; and learning on the information processes. Using the elementary adaptive selection process as a framework, we will, in this part, present an information processing theory of human verbal learning.

Our concern shall be with an elementary form of human learning--the acquisition by the human brain of information about objects in the environment and their properties. Our emphasis will be on verbal learning, with specific reference to the rote memorization of items on a serial list. We will deal with some of the basic functions which the brain performs; e.g., discriminating among the objects "out there," generating a response to a stimulus, recalling information previously learned. Functions of this type are seemingly going on all the time in the human brain, forming the hard core of all mental activity.

Our theory is an information processing theory, and its precise description is given in the language of a digital

computer. Although Part 3 develops this description, we choose to inject a few words about information processing models at this point.

2.1 THEORIES OF BRAIN FUNCTIONS AT THE INFORMATION PROCESSING LEVEL

The human brain is a biological mechanism, whose fundamental components are neurons and other cells, and whose operation at some basic neurological level is chemical or electrical in nature. No one knows yet precisely how the brain functions at this micro-level. There have been proposed, at a slightly higher level, theories of neural organization; e.g., Hebb's theory of cell assemblies (Hebb, 1949). We believe that there exists a useful level of theorizing about human behavior above the level of the organization of neurons; namely, a level at which information symbols are processed, organized, and stored. We shall refer to this level as the information processing level.*

A model framed at this level takes as its basic unit the information symbol (essentially a patterned collection of bits) which represents the brain's internal representation

*There is no requirement on a scientific theory that specifies the level of detail at which the theory must give an adequate explanation of the world. How far into the microstructure of a system one wishes to probe is determined strictly by the range of phenomena one is trying to explain and predict. Classical thermodynamics treats pressure as a fundamental entity manipulated by the equations of classical theory, ignoring molecular motion causing pressure, which is treated statistically at another level of detail by statistical mechanics. There is nothing inherently incomplete about classical thermodynamics. It simply does not explain a range of observations; e.g., the empirical Van der Wall's gas equation, which are handled by the micro-theory.

of sensory and verbal data; it postulates the existence of internal processes for organizing such information into behavior, and for storing information symbols by association with other symbols in one or more kinds of memory.

2.2 ROTE LEARNING TASKS

Our information processing model of verbal learning will be developed in the context of the experimental rote learning task. In this section, we briefly summarize the nature of this task.

Much of the experimental literature on verbal learning has dealt with rote learning, or rote memorization. The two methods most often used to study rote memorization are the serial-anticipation method, and the method of paired associates. These tasks are described as follows in the Handbook of Experimental Psychology (Chapter 15, Hilgard, "Methods and Procedures in the Study of Learning"):

"The serial-anticipation method. The chief laboratory method for the study of lists of items to be learned by rote memorization is that known as the serial-anticipation method, developed by Finkenbinder (1913) from Ebbinghaus' method of prompting. Items are presented one after the other, only one exposed at a time. The exposed item serves as the cue to the one that follows. It also serves to confirm or to correct the subject's anticipations. Because the subject announces his anticipation for each item, the experimenter can record the exact response and obtain not only the rate of memorization

but also the nature and number of the errors made."

"The method of paired associates. The task presented the subject in the method of paired associates is that of learning a series of discrete pairs so that the first member of the pair (stimulus member) will come to elicit the second member of the pair (response member). The learning of which pair follows which is not called for, and, in fact the order is varied in the preferred experimental arrangement so that serial position effects are counteracted....In order to obtain scores from the learning of paired associates similar to those from the serial-anticipation method, the memory drum is so arranged that the stimulus item appears in the aperture for a time before the response item appears. This permits the subject to announce the response item before it comes, so that the response may be recorded exactly as in the other method."

Needless to say, a great many variations of these two basic methods have been employed, involving method of responding, mode of responding, presentation rates, distribution of practice, learning criteria, etc.

The items of which the lists are comprised are either nonsense materials or meaningful materials. Nonsense materials are generally nonsense syllables, two consonants with a vowel in between. Meaningful materials are generally ordinary, familiar words of the language (nouns, verbs, adjectives, etc.)

Other experiments in which a subject learns two or more lists of items have been used to study proactive and retroactive inhibition (Britt, 1935, Swenson, 1941). List A is learned to a criterion, then list B is learned to a criterion. List A is tested again, and any decrement in the number of correct responses is attributed to "retroactive inhibition." "Proactive inhibition" is studied as follows:

Experimental Group	Control Group
Learn List A	No Learning of A
Learn List B	Learn List B
Retest List B	Retest List B

Any decrement in the performance on B after having previously learned A is attributed to "proactive inhibition."

The interested reader will find many more rote learning experiments described in ample detail in standard reference works for verbal learning phenomena (Handbook of Experimental Psychology, 1951; Woodworth and Schlosberg, 1956; McGeoch and Irion, 1952).

2.3 A MODEL FOR ROTE LEARNING

In this section, we present an information processing model for the acquisition of verbal materials in the rote learning task. The basic characteristics of the model are those of the elementary adaptive selection process, which was described in Part 1.

2.3.1 Macroprocesses and Microprocesses

The model is organized into two parts which are

conceptually distinct and which perform different functions in the overall learning process. We call these: macroprocesses and microprocesses.

The macroprocesses organize the total learning task and direct the attention of the learning organism.

The microprocesses encompass the processing which takes place on a portion of the stimulus environment which has been selected for attention by the macro-processes. They enable the organism to discriminate among stimuli it encounters; to fixate in memory information about stimuli; to form associations between fixated material; etc.

The interaction between the microprocesses and the macroprocesses is this alone: a serial processing mechanism must share its limited time among all the parts of the learning process; but the basic microprocesses--recognition, fixation, etc.--require a definite (rather large) amount of processing time on an element before this element can be considered as having been learned;" therefore, in terms of the total demands on the organism, there must be a set of processes which organize the time-sharing in such a way that learning will be orderly and reasonably efficient; and this is the function of the macroprocesses.

2.3.2 Macroprocesses

Elsewhere, we have described research done with H. A. Simon on the macroprocesses of rote serial learning (Feigenbaum and Simon, 1959). The investigation concerned a well-known phenomenon, the serial position effect. When subjects learn a list of items by the serial-anticipation method, their curve of errors vs. serial position is, characteristically, bowed. Least errors are made at the front and rear ends of the list, the number of errors rising smoothly to a maximum at the item just beyond the middle of the list. McCrary and Hunter (1953) have shown that when errors are plotted on a percentage basis, virtually all serial position curves from a wide variety of rote learning experiments are essentially identical.

The McCrary and Hunter serial position curve was predicted accurately by a macroprocessing model embodying four simple postulates, which will be given below. The model was programmed for a computer, and in ordinary serial-anticipation experiments the program exhibited behavior substantially indistinguishable from that of human subjects in the same experiments. Microprocesses were purposely not considered or spelled out, but were accounted for by assigning to their total function a fixed amount of processing time (see Postulate M2). Thus they were factored out completely; yet the model

was able to predict the important and well-established serial position curve. This evidence suggests that the factorization of our learning model into macroprocesses and microprocesses is indeed justified, and that the postulates for the former can be presented independent of those for the latter.

2.3.3 Macroprocessing Postulates*

MO: Serial Mechanism.

The central processing mechanism operates serially and is capable of doing only one thing at a time. Thus, if many things demand processing activity from the central processing mechanism, they must share the total processing time available. This means that the total time required to memorize a collection of items, when there is no interaction among them, will be the sum of the times of the individual items.

M1: Unit Processing Time.

The fixation of an item on a serial list requires the execution of a sequence of information microprocesses that, for a given set of experimental conditions, requires

---

*The postulates given here represent shortened versions of those given in the Feigenbaum and Simon (1959) paper. Consequently, they may appear to the reader to be "loose" when compared with the microprocessing postulates to be presented in succeeding sections. For a detailed and precise statement of MO-M3, we refer the reader to the aforementioned paper on the serial position effect.

substantial processing time per item.

M2: Immediate Memory.

There exists in the central processing mechanism an immediate memory of very limited size capable of storing information temporarily; and all access to an item by the microprocesses must be through the immediate memory.

M3: Anchor Points

Items in a list which have unique features associated with them will be treated as "anchor points" in the learning process. Anchor points will be given attention (and thus learned) first; items immediately adjacent to anchor points will be attended to second; items adjacent to these, next; and so on, until all of the items are learned.

Summary of M0 - M3

M0 establishes a serial processer, capable of doing only one thing at a time; this creates a need for deciding the order in which items will be processed, i.e. an attention focus, and M3 establishes a mechanism for determining this order. M2 provides a temporary storage, while the processes in M1 are permanently fixating the item.

2.3.4 Microprocesses*

In the discussion above, we ignored all consideration of the processes which act on an individual item (or pair of items) by collecting their total activity into a unit processing time. Now we wish to break down this activity into its component processes, and present postulates and discussion for each.

2.3.4.1 Selection Process

D2: Selection Process.

Information about external objects exists in the central processer as codes (as defined in Part 1), and images. An image is defined as a partial or total copy of a code. Images are organized in a memory in association with codes by means of the selection net structure discussed in Part 1. When codes are presented, their associated images are retrieved by the elementary selection process described in Part 1. Figure 2.1 represents a precise statement of the selection process, D2.

Discussion

Fixation ultimately results in the storage of an item in the memory of the processer. The

---

*Each of the microprocesses is given its precise statement as a program for an IPL computer. The details of the computer programs are given in Part 3. To identify the microprocessing postulates with these programs and thus insure continuity, we have given each postulate the same designation as its program counterpart. Hence, the names D2, F0, F1, D8, etc. Where no one-to-one correspondence exists, we have simply chosen a convenient designation, as F1a, R, etc.

D2. Obtain code for sorting.

Set up tracking list to record path taken through the discrimination net (track corresponds to the "diminished code" for the object in the net).

*1 Pick up the next symbol in the discrimination net.

Is it a testing node or a terminal point of the net?

Testing node	Terminal Point
Obtain and execute test	Obtain dictionary entry
On a minus result, obtain the branch right symbol.	Find image
On a plus result, obtain the branch right symbol.	Present image as output
Record the result on the tracking list.	
Return to *1	

Figure 2.1. Flow Chart for D2, Sorting Process

memory structure postulated by D2 is the tree structure of tests and alternatives presented previously. The codes used as argument to the selection process contain characteristics of external objects--as presented, say, by a perception process capable of capturing such characteristics in code. The images of these coded external objects are the internal representations of these objects. For purposes of simplicity of this model, they are assumed to be simple copies of part or all of the code for an object. The presentation of the code to the selection process results in the selection of the code's image.

To summarize, stored images of external objects are differentiated from each other on the basis of their position in a selection net, which is equivalent to saying that the system discriminates among images on the basis of information contained in the codes associated with the images. The discrimination is performed by the selection processes.

2.3.4.2 Discrimination Learning Process

In Part 1 we saw that the elaboration of the set of alternatives in a selection net usually involved the growing of branches of the tree-structure to accomodate the new alternatives.

The concept of a confusion at a terminal point of the net was developed. It was shown that confusion of items arose when two items contained the same information tested along a given path through the net. To put this in another way, two similar items, which may be somewhat different as far as their codes are concerned, may be confused because the discrimination net lacks sufficient testing power to catch the differences that exist. We have seen what corrective action may be taken to resolve the confusion: the codes of the two items are matched; differences are extracted; tests are constructed to capitalize on these differences; these tests are added to the net at the point of confusion. Such considerations lead directly to postulate F1.

F1: Discrimination Learning Process.

In the learning of a new item, it is necessary that a unique terminal in the discrimination net be provided for the storage of the image of the item. If the unique terminal happens to exist, the image of the item may be created and stored there; if not, a failure-to-discriminate occurs, which is resolved as follows: the code of the item to be learned is matched with the image already stored at the point of confusion. A difference (or differences)

is extracted. A test (or tests) on this difference (or differences) is created and the particular branch is grown at the point of confusion by adding these tests. At the termination of the discrimination learning process, the item just learned is discriminated from all other items previously learned. A precise description of the F1 process is given in Figure 2.2. A simple example of the action of F1 is given in Figure 2.3.

Discussion

The real meaning behind postulate F1 is to be found in a consideration of the discrimination capability which the system has in its net at any particular time. Roughly speaking, the net contains more information-decoding capacity than it needs at the time to distinguish among the items it already has learned (i.e., the images it already has stored.) Discrimination learning, however, is an involved and relatively expensive process (time-wise). To eliminate the necessity of performing the branch-growing process each time a new item is to be learned, the system may adopt a strategy of systematically introducing some redundancy into the net.

F1a: Adding Redundancy to the Selection Net

To circumvent the requirement that the branch

F1. Obtain Code for item
Apply D2

Image Found	Null Image
Match code to image, extracting differences.	obtain image from code enter image at terminal terminate.

Generate code positions for noticing differences (from noticing order)

Find bits in these positions from code and image.

Compare bits

Record any difference on list of differences.

Present list of differences.

Test for no differences, and terminate if no differences exist.

If differences found,

Create a test corresponding to each difference.

Create a subnet of these tests.

Insert subnet into net at terminal of confusion.

Old item is placed at "minus" branch of first new test.

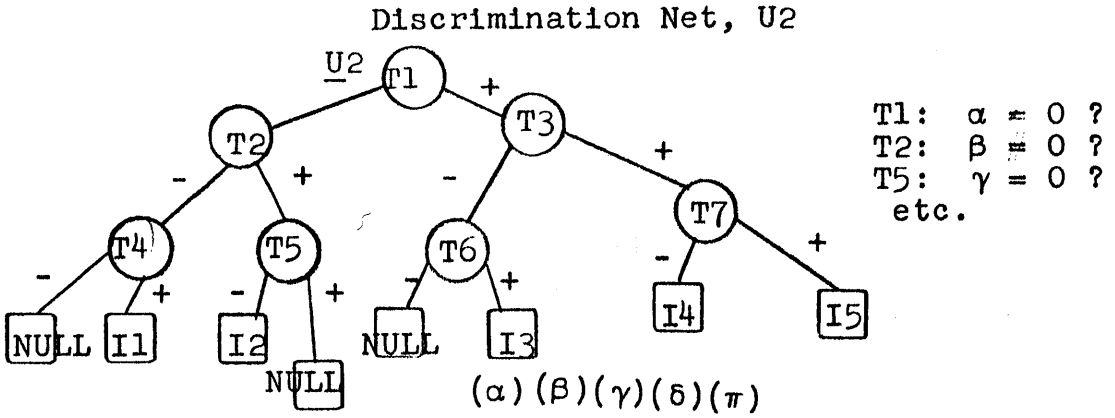
Other branches are given null images.

Execute F1.

Figure 2.2. Flow Chart of F1, Discrimination Learning Process

FI APPLIED TO LEARN NEW ITEM X IN NET U2

Before F1(X):



Code for New Item X: 1 0 1 0 1
Image, I2 : 1 0 1 1 0

Noticing Order : α, β, γ, π, δ.

During F1(X):

Confusion at terminal of Image I2.
One difference extracted (number is parameter controlled):
$X_{\pi} = 1$
(No differences found at α, β, γ)
Test Created: T8: π = 1?

After F1(X):

U2 remains unchanged except for:

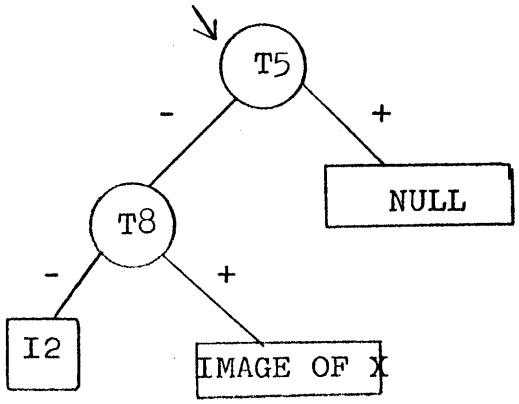


Figure 2.3 Highly Simplified Schematic Representation of Action of F1, Discrimination Learning Process

growing be performed each time a new item is learned, "empty" storage positions are systematically added to the net. This is accomplished by the procedure of extracting more than one difference between confused items, and introducing tests on the redundant differences into the net.

It may happen that the system is called upon to learn an item which it has already learned. In this case no further processing is necessary. The condition is recognized easily. The code of the item is matched for identity with the image selected by it. If they match in all the informational dimensions available (the code may contain more information than the image), it may be assumed that they represent one and the same object. Of course, this is a "heuristic," and subject to obvious failure if the amount of information available in the image is limited.

If differences between codes and images are to be extracted by the F1 process, they must be noticed in some order, just as the attention focus must notice items on a serial list in some order (postulate M3). The notion of a noticing order for codes was developed in Part 1. The ideas therein presented lead to the following postulate:

F1b: Noticing Order.

In the F1 process, the scanning of codes to be matched is governed by a noticing order, which is a list (or set of lists) containing, in order, the designations of code position(s) to be examined for differences. When a difference is found at a code position, that position is promoted in the noticing order, increasing its priority in the noticing order.

The postulate F1b represents a piece of the system's learning which enables it to adapt for efficient processing as the nature of the stimulus environment changes.

The implications of F1 will be drawn out in detail in Part 4 where the model is used in the simulation of rote learning experiments. Briefly, though, it may be said that F1 provides the machinery whereby the various items on a serial list, or those in associate pairs, are individually distinguished from the other members of the set--indeed, whereby any new stimulus item to be learned is distinguished from others of its class already learned. One of the consequences of F1 is that the greater the similarity of the syllables or words on a serial list or list of pairs, the more difficult will be the learning of the list (i.e., the more time the

learning will take.)

2.3.4.3 Association Learning Process

We have so far described a memory-organization system; an information retrieval process; and a discrimination process for adding new items to a memory. We will now deal with the problem of building internal associations between images in such a way that one image will become the response to another image.

In the discussion of associative connections between alternatives of the selection net given in Part 2, the concept of the cue code was developed. This concept is adopted to frame the following postulate defining a stimulus-response relationship between two images in a selection net.

F2: Stimulus-Response Association.

If two items, X and Y, which have already been learned in the discrimination net by the F1 process, are presented, and it is required that Y be made the response to X, the association learning process, F2, stores with the image of X a cue code for Y. The cue code consists of a subset of the information in the code of Y. The F2 process determines the size of the subset, by trial and error, to be that quantity of code information which will result in the selection of Y if the cue code is

used as the input to the selection process, D2. The precise description of F2 is given in Figure 2.4.

Discussion

At any given moment, the serial processer is very little concerned with the total learning task (e.g., the entire list of items). Its attention is focused on a single stimulus-response pair. Thus, our association theory is at odds with Hull's position, (Hull, et.al, 1940) which holds that a number of stimulus traces of various intensities can coexist in the organism during a learning trial and that a new stimulus, coincident in time with the set of lingering traces of old stimuli, may become associated with these stimuli. In our view, the system simply does not have available, at the moment of association, access to or immediate knowledge of remote items, and therefore cannot build remote links.

What the system does have available immediately are the codes for the two items to be put in S-R relation. As far as the system is concerned, these two items differ only in that they have different codes, and occupy, thereby, different places in the selection net. But the system has a way of "getting to" these places in the net at any time, namely through the selection process and the codes.

F2 Obtain codes of stimulus item (X) and response item (Y).

Apply D2 to X.

*1 Store a subset of code for Y with Image of X as cue code for Y.

Apply D2 to cue code for Y.

Match resulting image with Y for identity

If identical, terminate.

If not identical, return to *1.

Figure 2.4. Flow Chart for F2, Association Process

It is natural for the system to "retrieve" a response item by storing some code information about the response with the stimulus item.*

Let us take a concrete example. A list of associate pairs to be learned has on it the pair JAM-RIC. Let us assume that F1 has discriminated these items on the basis of their differing first letter. D2 finds the image of JAM. F2 stores the letter R with this image (in its coded form, of course) as the cue code for RIC. F2 then feeds this cue code to D2 as a trial to find out if this information is sufficient to gain access to RIC. Under our assumptions, it is sufficient; the association is complete.**

The association just formed is indirect through the discrimination net, and precarious. For what

---

*To store all of the response code with the stimulus item will usually turn out to be a waste of processing time, for codes are normally highly redundant, their information content far exceeding the decoding capability of the net.

**At this point it may be asked: why are response codes not stored directly, in toto, with the stimulus image? This is a real alternative, not to be lightly dismissed, and it seems likely that only empirical tests of the model will be able to distinguish the better of the alternatives. On a priori grounds, however, the indirect connection hypothesis, as embodied in F2, seems more reasonable. The very familiar phenomenon of response generalization seems to indicate that response items must be discriminated from other response items (and from stimulus items). This would not be captured by the alternate hypothesis.

happens to the S-R link if for some reason the branches of the net are grown (by F1) at various places? It may happen that RIC is repositioned to some place several levels below its original position by tests which scan second letters or third letters. Clearly, the information in the cue code, which was previously sufficient to select the correct response, RIC, is no longer sufficient.*

D8: Restoring an S-R Association

When a link has been previously successfully formed, and a response failure occurs, the D8 process will add additional information to the cue code, sufficient to re-establish the link. D8, of course, can only do this when both stimulus item and response item are available, as would occur in the normal sequence of events during the repeated presentations of a serial list in a rote learning experiment.

2.3.4.4 The Learning Sequence

FO: Control of the Order of Microprocessing Events

If X and Y are a pair of items in immediate memory, which may or may not have been previously memorized, and if these items are to be put in such a relationship that the presentation of X will evoke

---

*This contingency is both interesting and important, for it constitutes the only "forgetting" of which this model is capable. An interesting question is: are humans capable of any other? Forgetting, in this model, must be viewed not as the destruction or loss of information, but as the mislocation of information in the discrimination net. See Part 4 for an extended discussion of this point.

Y, then the following learning process will accomplish this:

Apply F1 to X
Apply F1 to Y
Apply F2 to X (stimulus) and Y (response).

2.3.4.5 Responding Processes

2.3.4.5.1 Processes which Produce Responses

To produce responses to the environment, any behaving system will have to have learned the actual physical responses in the set which it will be called upon to make. A child can not respond verbally until he has been taught the lip, tongue, and vocal chord combinations for each of the phonemes of the spoken alphabet, or the finger movements of calligraphy for the written alphabet. These are physical responses.

We have postulated that verbal responses are processed centrally in the form of images, or internal coded representations. Therefore, as a second requirement, there must be a process which maps images onto possible physical responses. And, of course, there must be a process for selecting the appropriate response to be generated to the environment.

R: Responder.

If a stimulus item X is presented, and the response to X is to be selected, the following

process is called for:

> Apply D2 to X, to obtain image of X (If no image, no response selected) Obtain from image of X the cue code for Y. Apply D2 to cue code for Y, to obtain image of Y. Image of Y is the selected response. If no cue code for a response is found, no response selected.

Response generation is the process whereby a response image is transformed into a physical response. Response generation is just another selection process. Response images are codes. If such an image is sorted by D2 in a discrimination net of physical responses, access is gained to the elements of the physical response.

D1: Response generation

Responses are generated by applying D2, one or more times, to a response image in a selection net of learned physical responses.

2.3.4.5.2 Modes of Responding and Perceiving

Human beings have various sensory devices for perceiving the environment. They also have various ways of emitting responses to the environment. Our theoretical position holds that internal representations of objects (images) are derived from codes of these objects; and that these codes are obtained by perceptual processes which scan and encode characteristics

of objects. There is no reason to believe that any given stimulus has only one internal representation. It may have a number, depending on the history of how it has been perceived in the experience of the organism. In particular, there may be an image of the object corresponding to each of the various sense modes.

Let us take an example. A translator scans text in a foreign language and stops at an unfamiliar word. Not only does the meaning escape him, but the word may not even be recognizable to him. He pronounces the word, however. Suddenly it is familiar, and the meaning "pops into mind." What may have happened is this: In the selection net for the words of the foreign language, there was no information stored at the branch corresponding to the "visual" code of the word. By pronouncing the word, (which may have been the way it was learned) he allowed a recoding to be accomplished into a "sound" mode. Associated with this new code, there was information stored, and this information led to a retrieval of the meaning.

There is no reason for assuming that an image in one mode should be anything like the image of the same object in another mode. There is

also no reason to assume that the same information about the object is stored with each image. In our theory, the codes in different modes must be regarded as separate entities, though they may be associated with each other by the ordinary kinds of associational links. It is possible, though this is just conjecture at the moment, that most of the internal processing humans engage in is done with "sound" codes; the reason for this may be that we as human beings make such great use, from early childhood, of word representations for objects, and that most of our references to external objects is via speech. An interesting discussion of imagery and sense modalities appears in a recent paper by Newell, Shaw, and Simon (1958).

Analagously, physical responses must exist in various modes. Corresponding to each mode there may be a selection net of physical responses for responding in that mode. However, if different nets, corresponding to different sense modes, are to exist, there must be a flexible method of associating images of the same object in various modes. The association processes already described will handle these translation problems.

2.4 SUMMARY OF THE POSTULATES

Macroprocessing Postulates

M0: Serial Mechanism. Capable of doing only one thing at a time, needs to time-share among its information processes.

M1: Unit Processing Time. Processing an item requires a substantial amount of processing time, used up by the micro-processes.

M2: Immediate Memory. There exists a fast-access, small size immediate memory, through which the microprocesses obtain their information.

M3: Anchor Points. Items with features of uniqueness will be attended to first for learning. Once these anchor points are established, attention will be given to items adjacent to those already learned.

Microprocessing Postulates

D2: Selection Process. There exists an elementary selection process (defined in Part 1) which uses codes (coded representations of external objects) to select images in a discrimination net.

F1: Discrimination Learning Process. Learning an item in the discrimination net consists of finding a unique terminal in the net at which to store the image, or creating such a terminal by growing the net at a point of confusion; i.e., adding discriminatory tests at that point.

F1a: Addition of Redundancy. Redundant tests are systematically added to the net to preclude the necessity of growing the net each time a new item is to be processed by F1.

F1b: Noticing Order. The scanning of codes for differences is controlled by a noticing order. The noticing order is updated on the basis of experience with finding differences.

F2: Stimulus-Response Association Learning Process. Two items, X(the Stimulus) and Y(the Response), are linked together in a stimulus-response relationship by storing with the image of X a cue code for Y; i.e., enough of the code for Y, so that D2 can use this cue code to select the image of Y. The quantity of information in the cue code is determined by trial and error.

D8: Restoration of S-R Link. When a link was previously successfully formed, and a response failure occurs, the D8 process will add information to the cue code, sufficient to re-establish the link.

F0: Control of Microprocessing Sequence. If X (stimulus) and Y (response) are a pair of items presented by the macroprocesses to the microprocesses for learning, apply F1 to X, apply F1 to Y, apply F2 to X and Y.

R: Responder. If stimulus item X is presented, and the response to X is to be selected, apply D2 to X, obtain cue code from image of X, apply D2 to cue code, obtaining image of response Y.

D1: Response generation. Physical responses are generated by applying D2 to a response image in a discrimination net of learned physical responses.

2.5 CONCLUSION

In this chapter, we have exhibited thirteen postulates for an information processing theory of rote serial learning. The postulates were shaped out of the processes and structures of the elementary adaptive selection model and represent a direct application of the more general theory given in Part 1. Essentially, all of the behavioral characteristics of the system stem from the properties of: a) the concept of the serial information processes, b) the use of internal codes for external objects, and c) the selection net as an organization of memory.

As presented up to this point, the postulates are not yet completely operational. For better or for worse, they are given their ultimate precise specification in a computer program called EPAM. EPAM I, programmed for the IBM 650 computer over a year ago, represented a precise statement of the macroprocesses. EPAM II, in the IPL-V computer language, is a precise statement

of the microprocesses. In Part 3, we endeavor to explain EPAM II; any theoretical points which remain vague at this point will, we hope, be illuminated by the next Part.

PART 3

EPAM II: Computer Program for an Information Processing Theory of Human Verbal Learning

3.0 INTRODUCTION

In Part 2 we drew the general outlines of an information processing theory of verbal learning behavior. Computer programs, counterparts of the postulates, have been written to allow a digital computer to carry out the information processing called for by the theory. The purpose of this chapter is to present the complete program for EPAM II, along with enough exposition to make the program intelligible to someone acquainted with simple programming in the computer language IPL-V. Sections 3.1-3.5 serve as a guide to the program given in section 3.6.

3.1 LANGUAGE SYSTEM

A few notes about IPL-V may be helpful. IPL-V is an interpretive language developed to facilitate the use of computers for symbol-manipulating tasks. Thus it contrasts with the currently popular trend toward algebraic languages for computers. Its "atoms" of information are not numbers, but symbols, and its processes deal with the organization, transfer, comparison, marking, etc. of these symbols. Its "molecules" are lists and list structures--ordered collections of symbols. Lists and list structures ordinarily form the domain over which IPL processes operate.

IPL processes represent the philosophy of subroutine hierarchies pushed to the extreme. Any symbol may be the subroutine name of any other set of symbols (and this set may include the symbol itself, if the subroutine is recursively defined). The IPL basic interpretive cycle examines all symbols as they come up for execution. If the symbol is the name of a primitive process, the intepreter executes it in machine language. If not, the interpreter descends into the subroutine, executing the instructions therein. There is virtually no limit to the depth of subroutining obtainable in this fashion.

3.2 BEHAVIOR SYSTEM AND ENVIRONMENTS

The total code currently written for EPAM can be conceptualized in two distinct pieces. On the one hand, there is a set of processes and structures which behave in the presence of stimuli. On the other hand there is a set of data list structures representing various environments in which the system is asked to behave. The two pieces are distinct because, so long as stimuli in the environments are encoded into a standard format expected by the behavior system, the system will process stimuli from any environment. In other words, the stimuli are essentially data for a general purpose piece of machinery, which is the behavior system.

3.3 STRUCTURES FOR EPAM ENVIRONMENTS

3.3.1 Encoding

The EPAM system, as presently coded, accepts stimuli which are binary-coded in a special format. Ideally, one would wish to expose EPAM to real environments, feeding the "raw" input data through an encoding process of the system, thereby producing binary codes for the selection and learning processes. However, EPAM does not as yet have perception programs. EPAM stimuli must be presented in coded form. It does not matter whether this code is completely arbitrary or corresponds to dimensions which are "real". In other words, if one, for example, wished to present the letters of the alphabet as stimuli to EPAM, he could choose some arbitrary coding scheme for categorizing the twenty six letters, or he could code such characteristics as "straight lines", "closed loops," "intersections," etc. It is important, however, that, whatever coding scheme is employed, it be used consistently throughout the pre-coding process.

3.3.2 Codes and Items

A code is a string of bits. For example, the code currently employed for the capital letter A in one of the EPAM environments is 10001100010111. More precisely, the structure of a code is an IPL-V list, each of whose elements is a bit: J4 for 0, and J3 for 1. Thus, in IPL-V, the code for the letter A is:

```
9-1   0000          (the local name is used because
      J3             the code is local to an item
      J4             list structure, which will
      J4             be explained)
      J4
      J3
      J3
       .
       .
       .
      J3    0000
```

The codes are the "atoms" of environmental representation. They are combined into more complex entities called items. An item is the unit of information presented to EPAM as a stimulus. An item is a list structure of codes. The precise format of an item is presented in section 3.6 with the identification ITEM. If, for example, we wish to represent the nonsense syllable XAP for presentation to EPAM, we could assemble the codes for X, for A, and for P into a list structure according to the conventions governing the format of an item, thereby constructing the stimulus for EPAM. (Provision is also made in the format of an item for including coded information about local context, peculiarities, etc., of any particular stimulus).

3.4 STRUCTURES OF THE EPAM SYSTEM

3.4.1 Selection Nets

In EPAM, internally stored information is organized by means of a selection net. This is a list structure of tests and images, and it is in a special format.

Images are found at terminal points of the net. Such a terminal point is a "pure symbol" (i.e., an empty list, with a description), whose description contains, as the value of attribute A1, the name of the dictionary entry under which the image can be found. (Image and dictionary will be defined and discussed later.) A terminal point is distinguished from a testing point by the value of attribute AO. This value is J3 for a terminal point, J4 for a testing point.

A testing point is a list of 3 symbols. The first symbol on the list is the name of a test to be applied. Upon execution, the test leaves a signal in H5, which is "branch minus" or "branch plus." The "branch minus" and "branch plus" symbols are the second and third items on the list. The description list of the testing point contains information which the test needs to locate a particular code characteristic; this information is given as the value of various attributes.

A typical selection net is the net for sorting capital letters, U1. In this net, ZO is the test performed at all the testing points. ZO finds the information it needs as the values of attributes A4 and A5 on the description list of the testing point.

Example: the following is a selection net satisfying the above conventions.

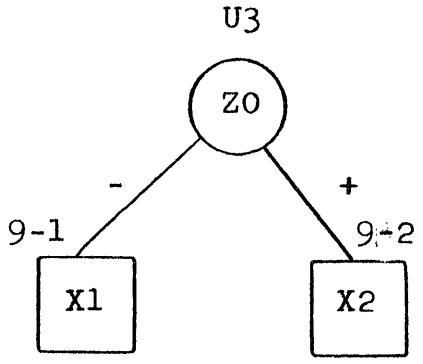
```
U3    9-0        U3 is the name of the net, also first testing
                 point
      Z0         Z0 is the test, using V(A4) and V(A5) of 9-0
      9-1        branch minus to 9-1
      9-2 0000   branch plus to 9-2

9-0   0000       description list of U3
      A0         Is this a testing point or a terminal point?
      J4         This is a testing point.
      A4         Which line of code in item does Z0 examine?
      N1         First line.
      A5         Which bit in that line of code?
      N8 0000    Eighth bit.

9-1   9-3 0000   9-1 is a terminal point
9-3   000
      A0
      J3         This is a terminal point
      A1         Where do I find the image stored here?
      X1 0000    In the dictionary, under heading X1.

9-2   9-4 0000   9-2 is a terminal point
9-4   0000
      A0
      J3         This is a terminal point
      A1
      X2 0000    Image under heading X2.
```

The tree-structure representation of this simple net is:



Selection nets, in general, are very much more complicated than this, as one can discover for himself by a careful diagramming of the net U1 presented in section 3.6.

A summary of the meaning of the attribute symbols used in selection nets is:

A0: designation, testing point or terminal point?
A1: location of the image in the dictionary
A4: line of code in an item, to be examined by general test Z0.
A5: bit in a code to be examined by general test Z0.

3.4.2 Images

The selection net organizes stored information. Access to this stored information has been provided at terminals of the net. The unit of internally-stored information is the image. The image represents an internal representation of a coded environmental stimulus. The simplest form which an image can take on is that of a copy of the coded stimulus. This attack was taken initially in EPAM II, but other kinds of images will undoubtedly be experimented with. The variations are many: images containing part (and not all) of the coded stimulus information; images obtained by translating the input code from one sensory mode to another; images which correspond not to the input stimulus but to some immediate response to this stimulus; and so on. However, for the model as now coded, images are exact stored copies of the input stimuli.

3.4.3 Responses

One of EPAM's primary functions is to respond to its environment. Internally, images may be linked together in association, so that the presentation of one will call

forth a second in response to it. But this is not yet an intelligible response to the environment. It is merely a collection of internal codes. What is needed is a repertory of actual physical responses that EPAM can make, and a method of mapping images onto this set of responses. A selection net is the appropriate mapping, and this is the function of net U1. The repertory of physical responses (capital letters of the alphabet) associated with net U1 are the L's (L1 to L26).

3.4.4 Dictionary

The dictionary provides an added piece of potential flexibility, which is currently not used to advantage in EPAM II. Because it has certain interesting implications for future models, it has been preserved. The dictionary is a pure symbol. On its description list are attribute-value pairs, corresponding to a dictionary entry and an image entered there. The symbol inside each dictionary entry is the name of the dictionary. Cell Y3 always holds the name of the dictionary.

Example:

Let X0 be the dictionary. Under entry X1 in dictionary X0 is the image I6. There are no other entries in the dictionary. The relevant cells contain the following symbols:

Y3 X0 0000

X0 9-0 0000
9-0 0000
X1
I6 0000

X1 X0 0000

3.4.5 Noticing Order

The process which notices differences between codes and images scans for differences on the basis of symbols in the noticing order. The noticing order consists of two lists, PO and Pl. PO gives the order in which the lines of code of an item are scanned. Pl gives the order in which the bits within a line are scanned. The symbols on PO and Pl are integer data terms.

Examples:

```
PO   0000

     N 1        first line
     N 3        third line
     N 2   0    second line

Pl   0000
     N 1        first bit
     N 2        second bit
     N 3        third bit
       :
       :
       :   0
```

3.5 PROCESSES OF THE EPAM SYSTEM

3.5.1 D2, Selection Process

The selection process, D2, accepts the following input:

(0): item to be decoded

(1): selection net in which item is to be decoded.

The output, (0), of D2 is the name of the selected image.

3.5.1.1 DO, The Net Interpreter

D2 uses DO to move through the selection net. DO is given the name of the net (0).

The summary of DO's operation is as follows: DO interrogates the designation for terminal points and testing points. At a testing point, it finds and executes the test, examines the output signal (in H5), finds the appropriate branch (minus or plus), and iterates. Upon reading a terminal point, it retrieves the appropriate image from the dictionary.

3.5.1.2 Tests: Z0 and Z1

Tests are active processes which examine a particular characteristic of a code, and leave an output signal in H5. Currently, there are only two broad classes of tests to be found in our nets. The class of Z0 tests interrogates a particular bit in a line of code and sets H5 + on finding J4, - on finding J3. In the class of Z1 tests, H5 is + on finding J3, - on finding J4. If Z0 or Z1 can not find the information they need, they set H5 randomly (with equal probability of + or -). Every instance of a Z0 or Z1 test in a net is accompanied by the presence of a pair of integers on the description list of the testing point. These integers designate line and bit to be interrogated. Thus, every instance of a Z0 or a Z1 is a "specific" test, and the system generates "specific" tests on differences as it needs them.

3.5.1.3 Tracking List

Each time DO cruises through a net, it keeps a record of the "branch right-branch left" track it takes. Before DO terminates, it assigns the track to the image selected, as the value of attribute A2. The origin of the tracking is assigned to the track as the value of attribute A3.

3.5.2 Learning Processes: FO, F1, F2

The central control of EPAM's internal processing is FO. Given a stimulus item (0) and a response item (1), FO adds each item to the selection net, and builds the cue-code link between stimulus and response.

F1, the discrimination learning process, adds an item (0) to a net (1), building a discriminating test structure where necessary. The item is used by D2 to select an image. If the image is non-null, and different from the item, D5 notes the differences, D7 creates a test structure to resolve the confusion, and D10 adds the test structure to the net.

F2, the association learning process, builds a cue-code link between a stimulus item (0) and a response item (1) in a net (2). The stimulus selects an image (using D2); the cue-code with this image is retrieved (D7) and used to select a "tentative" response image (using D2). If the response and "tentative" response image do not match, D8 adds some more of the response item's code to

the cue-code. If no cue code exists, F2 sets one up and uses D8 to build the cue-code.

3.5.3 Basic Functional Processes

3.5.3.1 D18, Notice Code Positions

D18 is a "scanner" which generates code positions to be examined for differences. It generates these in the noticing order, P0 (line) and P1 (bit).

3.5.3.2 D5, Produce Differences Between an Item and an Image

D5 produces a list of differences between an "incumbent" image (0) in the net, and a "challenger" item (1). The difference produced is in the format given in section 3.6 with the label DIFF. The maximum number of differences it will produce is parameter controlled by the data item in P2. D5 also updates the noticing order based on the differences found. For each difference, the particular line and bit at which the difference was noticed are promoted up one list position on the lists P0 and P1.

3.5.3.3 D9, Build Test Structure Using Differences

D9 creates a net structure using a list of differences (0). The terminal at which the original confusion took place has to be inserted into this new test structure, and is given as input (1). For each difference, a Z0 or Z1 test is set up, with appropriate A4 and A5 pointers. Currently, the

"terminal of confusion" (1) is inserted along the first "minus branch" of the new test structure. The name of the new net structure is given as output (0).

3.5.3.4 D10, Insert Test Structure into Net

D10 adds a net structure (0) created by D9 to a selection net. The testing point which is to point to the new subnet is given as (1), and the former "terminal of confusion" is given as (2). D10 erases the old terminal and inserts the name of the new subnet in place of the old terminal.

3.5.3.5 D17, Create Image

D17 creates an image from an item (0). Currently, D17 merely produces an identical copy of the item as the image.*

3.5.3.6 D7, Obtain the Cue-code Stored with an Image

Given the name of the image, D7 finds the cue-code as the value of attribute A8.*

3.5.3.7 D8, Add Information to Cue-code

D8, as presently coded, adds information about a response item (0) to the cue code of image (1)

---

*It is anticipated that as the model changes, D7 and D17 will become more complex processes. For D7, the changes will involve searching for cue-codes in different symbolic contexts (see 1.4.2). For D17, the changes will be in the direction of less information in the image.

line by line. That is, it adds not individual bits of an item, but whole lines of code. It determines what lines are already in the cue-code, and then adds the "next" line, as determined by the noticing order for lines, P0 (the same noticing order used by D18 to scan for differences). D8 adds only one new line each time it is used.

3.5.3.8 D15, Match Item and Image for Identity

D15 compares two lists in ITEM format, bit by bit, for identity. The result of the match is left in H5: + for identity, - for non-identity. D15 only reports non-identity when an actual discrepancy is noted; i.e., when information is present in one list and not in the other. This is treated as a "match-by-default."

3.5.3.9 D12, Test for no Image

D12 determines whether a selected image is one of the null images present in the net. The symbol used to stand for a null image is K10.

3.5.3.10 D1, Response Generator

D1 is the process which makes a physical response to the environment. Its input (0) is an image. D1 uses D0 to interpret the net of capital letters, U1. Assuming that each line of code in an image corresponds to one of the capital letters stored in the net U1, D1 will print out the image as a string of capital letters.

3.5.4 Summary of EPAM Processes

F0: Stimulus-Response Learning Control Process
Input (0) = stimulus item
(1) = response item
(2) = name of selection net

F1: Discrimination Learning Process
Input (0) = name of item
(1) = name of selection net

F2: Association Learning Process
Input Same as F0

D0: Selection Net Interpreter
Input (0) = name of net
Output (0) = name of selected image

D1: Response Generator
Input (0) = name of image

D2: Selection Process
Input (0) = name of item
(1) = name of selection net
Output (0) = name of selected image

D5: Produce a list of differences
Input (0) = incumbent item
(1) = challenger item
Output (0) = list of differences

D7: Find cue-code stored with image
Input (0) = name of image
Output (0) = name of cue code

D8: Build cue-code with image
Input (0) = name of response item
(1) = name of image

D9: Create discriminating test structure
Input (0) = list of differences
(1) = terminal of confusion
Output (0) = name of test structure created

D10: Add test structure to selection net
Input (0) = name of test structure
(1) = name of point of insertion
(2) = terminal of confusion

D12: Test for null image
Input (0) = name of image

D15: Match items or images for identity
Input (0) = an item or image
(1) = another item or image

D13: Noticing Generator
Output (0) = line to be scanned
(1) = bit in line to be interrogated

A1 - A8 utility routines for description lists. Also attributes.
E1 - E13 utility routines used by other processes

Z0: Test for a "plus" bit
Input 1W2 = Testing point of net
1Y1 = Name of item being decoded
Output Signal in H5

Z1: Test for a "minus" bit
Input and Output Same as Z0

3.6 EPAM Programs in IPL-V

The Appendix gives the complete code for EPAM II in IPL-V, including programs and data list structures. Programs and list structures are arranged alphabetically according to the identification and sequence numbers on the right side of the page.

PART 4

SOME IMPLICATIONS OF EPAM II IN ROTE LEARNING SITUATIONS

4.0 INTRODUCTION

The primary goal of this reserach has been to develop an information processing theory of human verbal learning. In Parts 2 and 3, we presented the hypotheses of the theory and their computer program counterparts. If the hypotheses have validity, there should be a number of verbal learning phenomena which the theory will explain.

At this stage of the research, we can not give a formal test of the theory.* The purpose of this Part is to present some of the more obvious implications of the EPAM model for rote serial learning. The behavior of the system is complex enough so that even "obvious" implications are not so obvious until the program is made to behave, even if only in simple situations. So that some of this behavior can be examined, the program has been simulated by hand, in rote learning tasks. The simulations and the results will be exhibited.

4.1 THE BEHAVIOR OF EPAM IN ROTE LEARNING EXPERIMENTS

In the experiments presented, EPAM was made to learn short lists of nonsense syllables, by the serial anticipation

---

*The only realistic way to reach the stage of formal testing is to use the IPL-V computer to generate the behavior of the EPAM system under various experimental conditions. At the time of this writing, the IPL-V computer is not yet fully completed. The analysis of this section must take place at a more general level than is ultimately desirable. Therefore, no claim about a formal test is made here.

method, and by the method of paired associates. For each experiment (4.1.1 and 4.1.2) the following information will be given: the experimental arrangement (including environmental conditions and initial machine states); the behavior of EPAM and the behavior characteristic of human subjects in such an experiment; and a discussion of the results.

4.1.1 Single List Experiments with Nonsense Syllables: Serial Anticipation and Paired Associates

4.1.1.1 Experimental Arrangement

4.1.1.1.1 Environmental Conditions

a) Method: Serial Anticipation (see 2.2)

Materials: The following six nonsense syllables:

LIST A

KAG
LUK
RIL
PEM
ROM
TIL

Criterion: one perfect run-through

b) Method: Paired Associates

Materials: The following three nonsense syllable pairs:

KAG-LUK
RIL-PEM
ROM-TIL

4.1.1.1.2 EPAM Initial State

1) Codes. Ordinarily syllables are presented to EPAM as items, consisting of a line of code

for each letter, and perhaps other coded information. For purposes of simplicity in this hand-simulation, we will act as though we were using codes, and as though these codes had certain properties. The use of actual codes for letters would involve us in an enormous amount of detail, deleterious to our purposes here.

2) Selection Net. To give as much illustration as we can of the growth of selection nets, we begin with a completely null net, U2.

3) Noticing Order. We assume a kind of "serial position effect" for the scanning of the individual syllables, so that the order of noticing differences in the letters of syllables is initially

First Letter

Third Letter

Second Letter

The maximum number of differences noticed (and tests added) each time the net is grown will be taken to be three.*

*The assumptions given above about pre-experimental states are more or less arbitrary, but we have satisfied ourselves that the qualitative behavior of the system, in which we are currently interested, does not depend in any sensitive manner on the precise form of these assumptions.

4.1.1.2 The Simulation

The entire simulation of verbal learning behavior in the single-list experiments is presented in Figures 4.1 and 4.2. These figures exhibit:

a) each major cycle of the internal processing, controlled by FO.

b) the selection net at the end of each cycle.

c) the ongoing "external" trial-by-trial behavior.

4.1.1.3 Results of the Simulation

In this section we discuss some features that the simulated behavior has in common with human rote learning. For each topic considered, we present a brief explanation of the phenomenon (subsection a), EPAM's behavior (subsection b), and the evidence about the human behavior (subsection c).

This treatement will be given to the following features of the behavior: disjunctive reaction time of subjects (4.1.1.3.1); stimulus and response generalization (4.1.1.3.2); effects of intralist similarity (4.1.1.3.3); oscillation (4.1.1.3.4); types of errors and their distribution (4.1.1.3.5); learning time for serial-anticipation vs. paired

associates learning (4.1.1.3.6); and, in a later section, retroactive inhibition (4.1.2.3). These topics are typical of the issues that have been raised in the study of human rote learning. We have selected these particular phenomena because we feel that something relevant can be said about them, even from the relatively crude results of the hand simulation. Of course, we will deal mostly with qualitative, rather than quantitative, aspects of behavior. A more complete analysis of rote serial learning will be made when an IPL-V computer becomes available.

4.1.1.3.1 Disjunctive Reaction Time

a. A subject is presented with one of n possible stimulus patterns and is required to perform one of k different responses. Stimuli and responses are familiar. The subject is instructed to perform as quickly and accurately as possible. Disjunctive reaction time is measured as the elapsed time between the reception of the stimulus and the beginning of

text continues on page 101

Figure 4.1: Processing of a List of 6 Nonsense Syllables by Method of Serial Anticipation

External Activity		Internal Processing	Comments
S	R Given		
Trial 1			
KAG	none*	FO (KAG, LUK)	
LUK	none*	F1 (KAG)	
RIL	none*	F1 (LUK)	
PEM	none*		1. T1, T2, T3 are first letter tests.
ROM	none*		○ = tests.
TIL	____	F2 (KAG, LUK)	[kag,1] = image of kag with cue code 1.
Trial 2			
KAG	LUK		[ ] = null images

*signifies response error

Selection Net at End of FO cycle shown above[1]

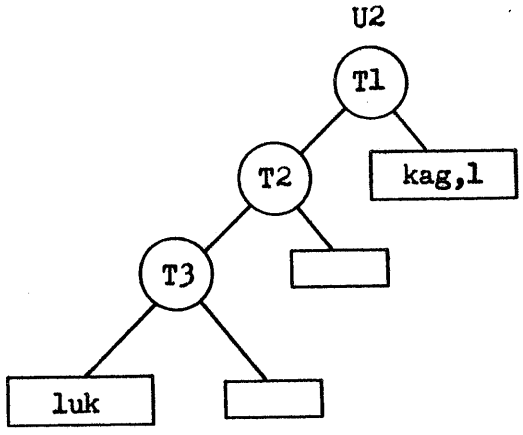


Figure 4.1: Processing of a List of 6 Nonsense Syllables by Method of Serial Anticipation (Continued)

External Activity		Internal Processing	Comments
S	R Given		
LUK	none*	F0 (LUK, RIL)	
RIL	none*	F1 (LUK)	
PEM	none*		
ROM	none*		2. T4, T5, T6 are first letter tests.
TIL		F1 (RIL)	
Trial 3			
		F2 (LUK,RIL)	
KAG	LUK		
LUK	RIL		
RIL	none*		
PEM	none*		

Selection Net at End of F0 cycle shown above[2]

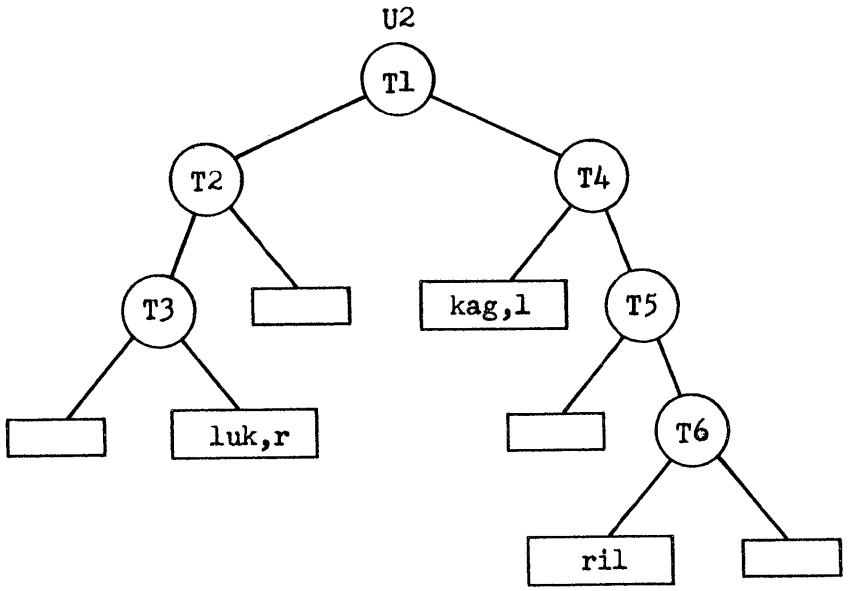


Figure 4.1: Processing of a List of 6 Nonsense Syllables by Method of Serial Anticipation (Continued)

External Activity		Internal Processing
S	R Given	
ROM	none*	
TIL	____	F0 (ROM, TIL)
		F1 (ROM)
Trial 4		
KAG	LUK	
LUK	none*[6]	F1 (TIL)
RIL	none*	
PEM	TIL*[7]	F2 (ROM, TIL)

Comments

3. T7, T8, T9 are third letter tests.

4. Noticing order updated.

5. T10, T11, T12 are third letter tests.

6. When cue code contains insufficient information, D2 selects randomly, in this case a null response.

7. PEM is confused with ROM and response to ROM is made.

Figure 4.1: Processing of a List of 6 Nonsense Syllables by Method of Serial Anticipation (Continued)

Selection Net at end of FO cycle shown on previous page.[3,4,5]

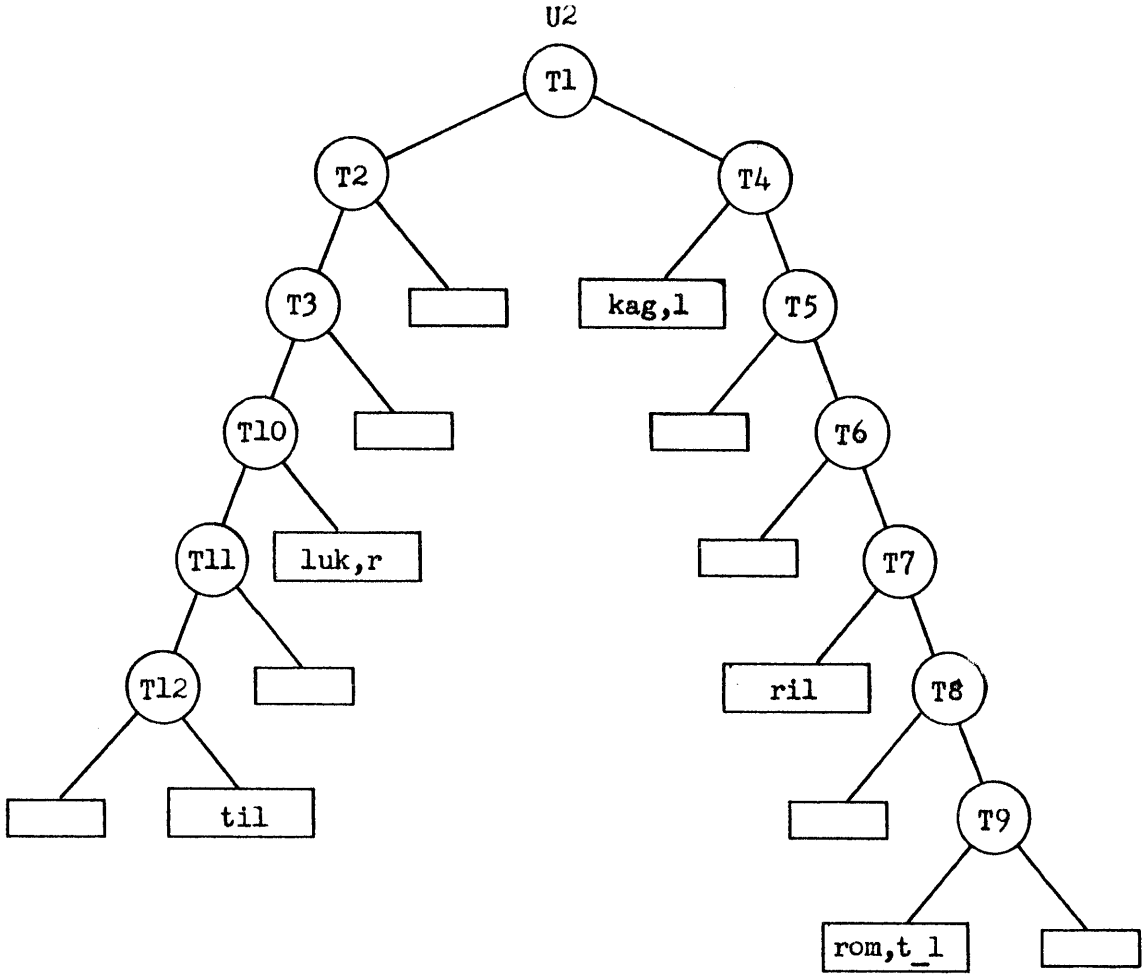


Figure 4.1: Processing of a List of 6 Nonsense Syllables by Method of Serial Anticipation (Continued)

External Activity		Internal Processing	Comments
S	R Given		
ROM	TIL	FO (PEM, ROM)	8. T13 is a first letter test.
TIL	___	F1 (PEM)	
Trial 5			9. Only one test is added, because for variety we assume only one difference found between R and P
KAG	LUK[12]	F1 (ROM)[9]	
RIL	none*[10]		10. LUK-RIL link re-established after error on trial 4.
PEM	none*[11]		
ROM	TIL	F2 (PEM, ROM)	11. PEM is discriminated but link is not yet established.
TIL	___		
Trial 6			12. KAG--LUK is precarious (not enough cue-code for LUK but by chance here the correct response was given (see trial 6).
KAG	none*		
LUK	RIL		
RIL	none*		
PEM	ROM		

Changes in Selection Net at end of this FO cycle

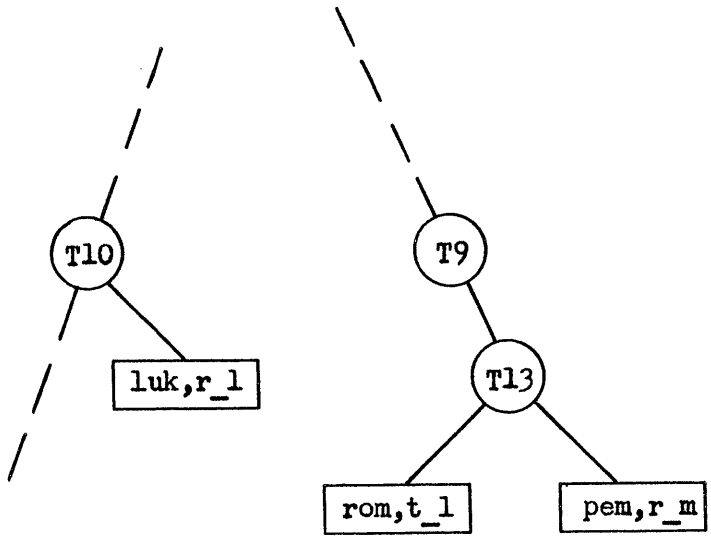


Figure 4.1: Processing of a List of 6 Nonsense Syllables by Method of Serial Anticipation (Continued)

External Activity		Internal Processing
S	R Given	
ROM	TIL	F0 (RIL, PEM)
TIL	___	F1 (RIL)
Trial 7		F1 (PEM)
KAG	LUK	F2 (RIL, PEM)
LUK	RIL	
RIL	PEM	
PEM	ROM	
ROM	TIL	
TIL	___	

Figure 4.1: Processing of a List of 6 Nonsense Syllables by Method of Serial Anticipation (Continued)

Final Selection Net for this Task

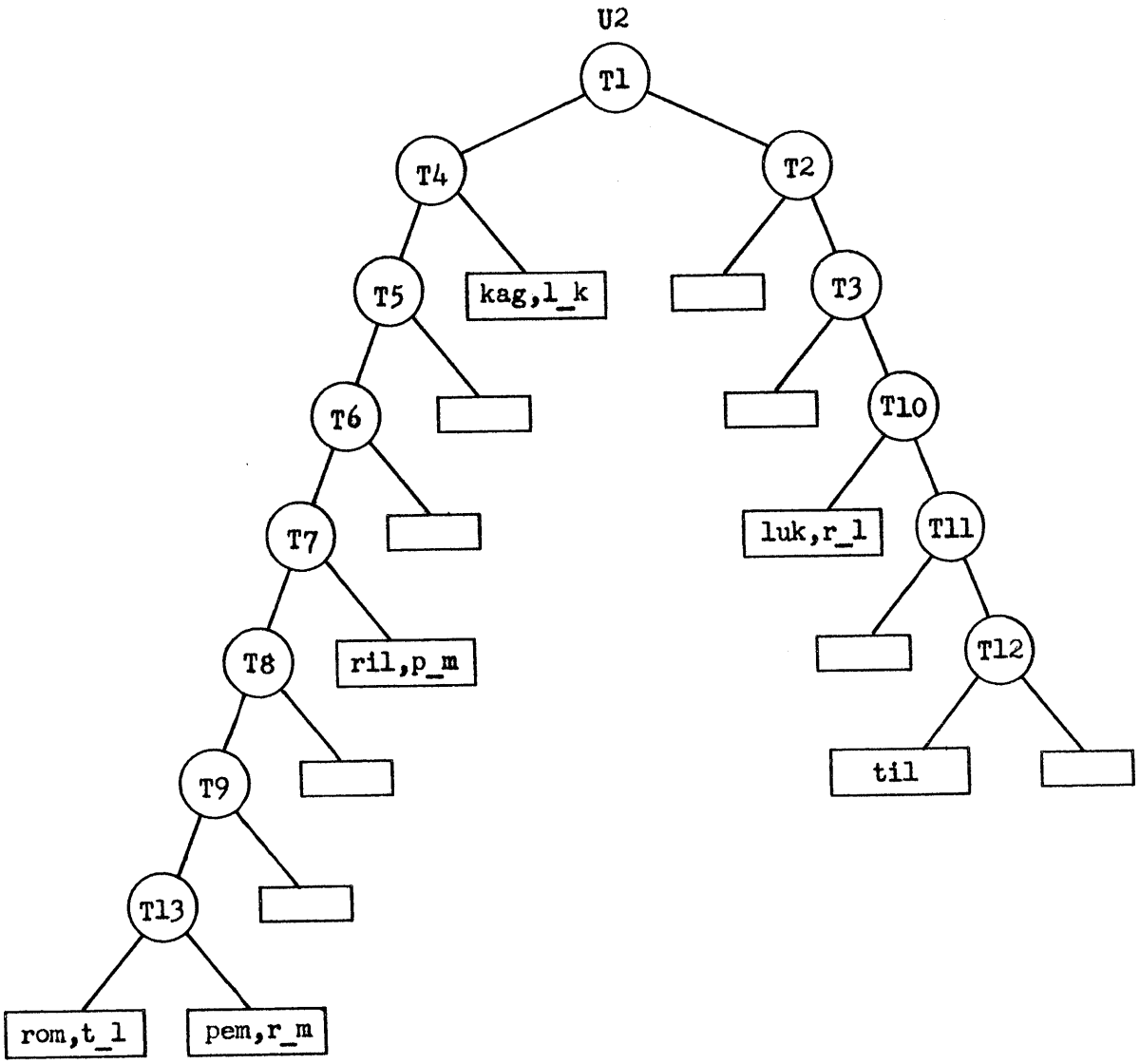


Figure 4.2: Processing of a List of 6 Nonsense Syllables By Method of Paired Associates

External Activity			Internal Processing	Comments
S	R	R Given		
	Trial 1			
KAG	LUK	none*	FO (KAG, LUK)	
RIL	PEM	none*	F1 (KAG)	
ROM	TIL	none*		
	Trial 2		F1 (LUK)	
KAG	LUK	LUK[1]		1. KAG and LUK are still immediately accessible in immediate memory.
RIL	PEM	none*		
ROM	TIL	none*	F2 (KAG,LUK)	
	Trial 3			
KAG	LUK	LUK		
RIL	PEM	LUK*		

* signifies response error

Selection Net at End of FO cycle shown above

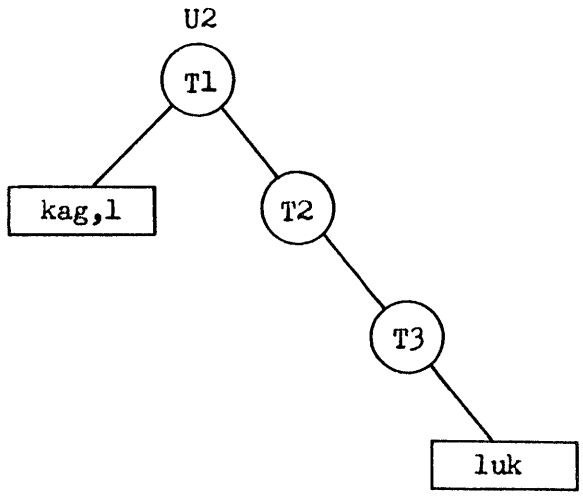


Figure 4.2: Processing of a List of 6 Nonsense Syllables By Method of Paired Associates (Continued)

External Activity			Internal Processing	Comments
S	R	R Given		
ROM	TIL	LUK*	FO (ROM, TIL)	
	Trial 4		F1 (ROM)	2. No oscillation here as in serial list. Noticing order is such that first letter tests distinguish LUK and TIL.
KAG	LUK	LUK		
RIL	PEM	none*	F1 (TIL)	
ROM	TIL	none*		
	Trial 5		F2 (ROM,TIL)	
KAG	LUK	LUK[2]		
RIL	PEM	TIL*[3]		3. Generalization error.

Selection Net at End of FO cycle shown above

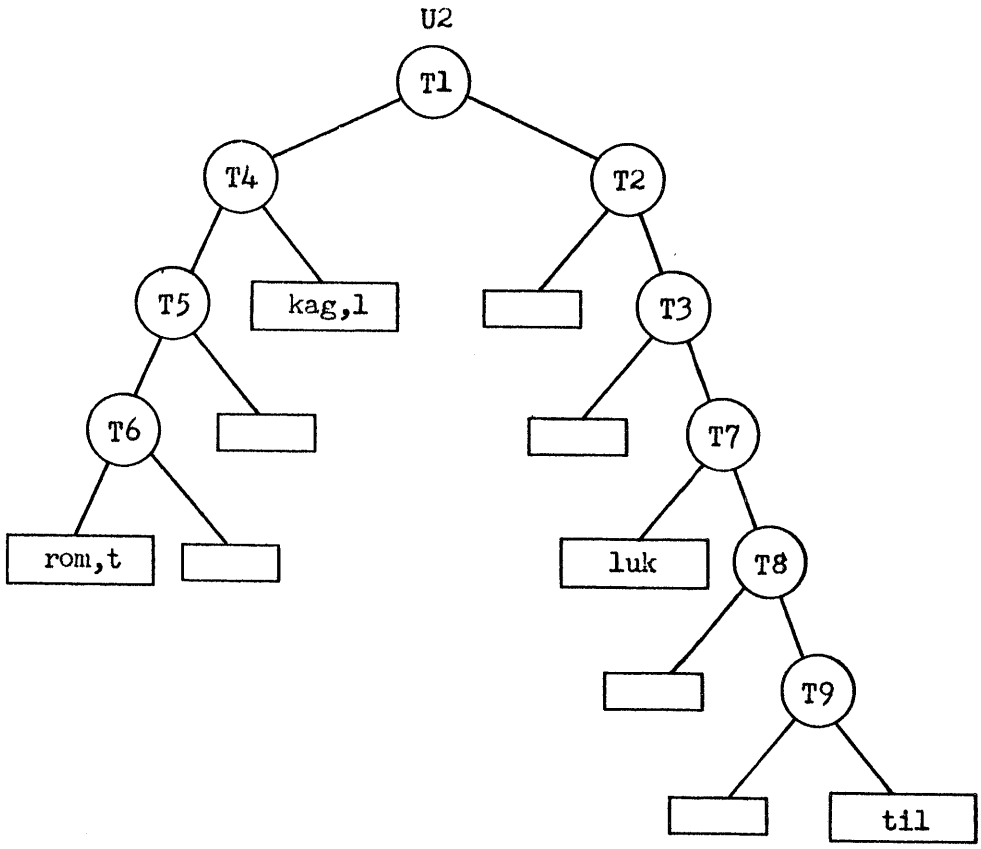
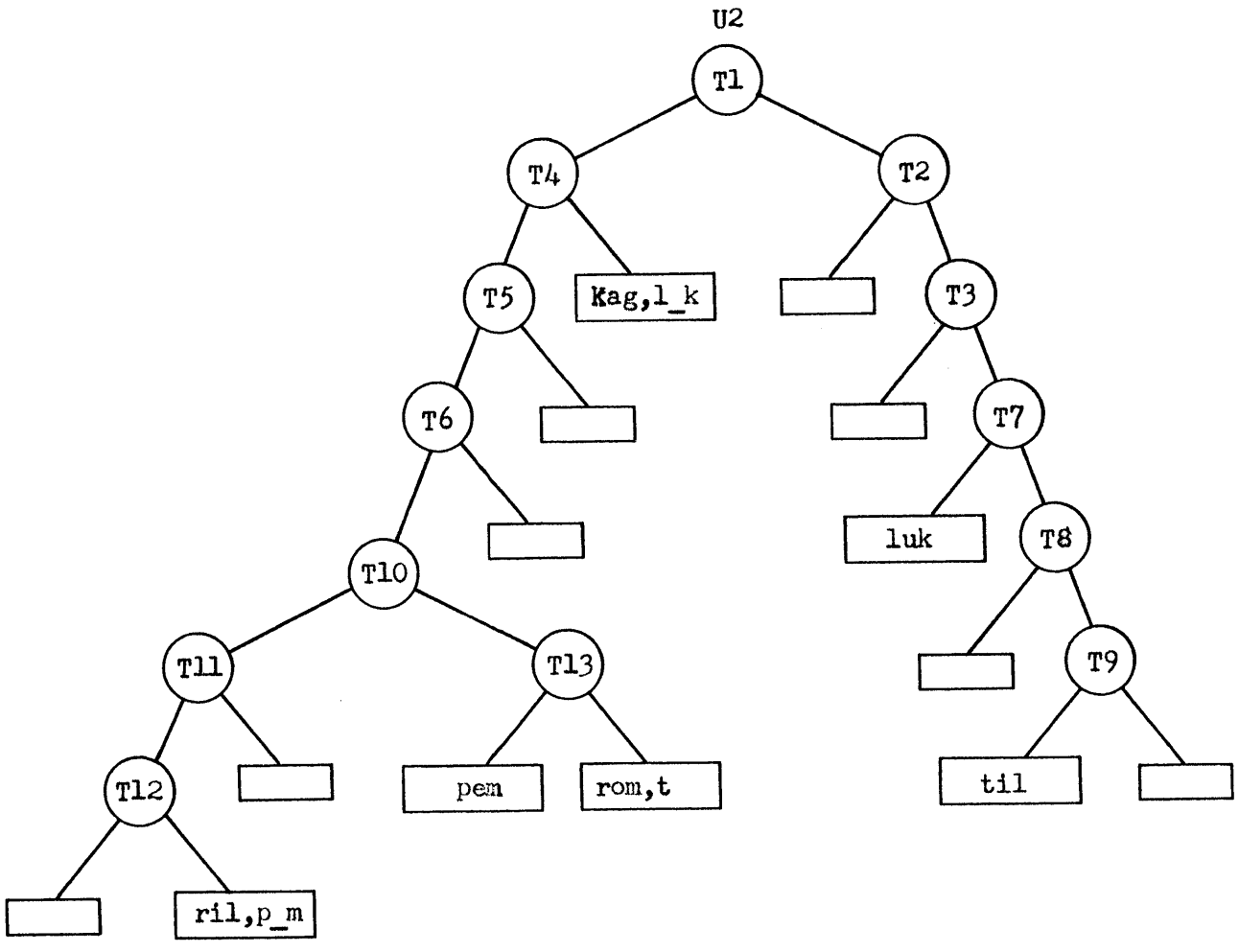


Figure 4.2: Processing of a List of 6 Nonsense Syllables By Method of Paired Associates (Continued)

External Activity			Internal Processing	Comments
S	R	R Given		
			F0 (RIL, PEM)	
ROM	TIL	TIL	F1 (RIL)	
	Trial 6			
KAG	LUK	LUK	F1 (PEM)	
RIL	PEM	none*		
ROM	TIL	TIL		4. First letter tests: T1, T2, T3, T4, T5, T6, T7, T8, T9, T13. Third letter tests: T10, T11, T12.
	Trial 7			
			F2 (RIL, PEM)	
KAG	LUK	LUK		
RIL	PEM	PEM		
ROM	TIL	TIL		

Final Selection Net for this Task[4]



the response.

b. For the EPAM system, disjunctive reaction time is approximately the following function of the number, n, of alternatives in the selection net:

$$D.R.T. = a + b \log_2 n$$

where a can be considered to be a constant "set up" time for the selection process (encoding, etc.), and b is the constant processing time per cycle of the selection process (i.e., per level in the tree structure). The dependence of D.R.T. on the $\log_2 n$ arises because in the selection net average number of levels and n are approximately related as: $2^{\text{no. of levels}} = \text{no. of alternatives}$.

c. In a number of experiments performed in the last decade, the dependence of disjunctive reaction time on the log of the number of alternatives has been generally substantiated (Hick, 1952; Crossman, 1953; Hyman, 1953). Bricker (1955), however, is careful to qualify these results by noting that they depend on the presence of "ideal" conditions (i.e., mainly simplicity of the stimulus and response material and the experimental design). Recently, Mowbray and Rhodes (1959) have contended that

the logarithmic dependence holds only for conditions in which no overlearning of the alternatives takes place.

4.1.1.3.2 Stimulus and Response Generalization

a. Stimulus generalization: a subject has learned that a response Y is to be made when a stimulus X is presented. At a later time, a different stimulus X'; similar to X, is presented and the subject responds with Y. Response generalization: a subject has learned response Y to stimulus X. If Y' is a response similiar to Y and in the set of responses being learned, the subject may make Y' as the response to X.*

b. There are four examples of stimulus generalization in the single-list experiments:

---

*In their discussion of generalization, Woodworth and Schlosberg (1954) have this to say in a footnote (p. 755-756):

> "This meaning of "generalization" is very different from the usual meaning which is often encountered in the literature of transfer, as when the meager transfer value of special training is attributed to the learner's failure to "generalize" from the special task to the broader application of the acquired ability. Generalization in this ordinary sense is an achievement, but in the Pavlovian sense it is no achievement but a primitive state of behavior, the only achievement being to advance out of this stage by aid of differential reinforcement. Pavlovian generalization might be called nondifferentiation or perhaps primitive generalization. "

Method	Trial	Stimulus	Wrong Response
Serial Anticipation	4	PEM	TIL
Paired Associates	3	RIL	LUK
Paired Associates	3	ROM	LUK
Paired Associates	5	RIL	TIL

There are no examples of response generalization in the single list experiments. There is one example of response generalization in the two-list experiment exhibited in section 4.1.2.2 and Table 4.2.*

Method	Stimulus	Wrong Response
Serial Anticipation	KAG	LEK

c. The phenomena of stimulus and response generalization in verbal learning are familiar ones. The fact that they do occur is incontestable. The most thorough study is that of E. J. Gibson s (1940, 1941, 1942). Of much interest, also, are the studies of generalization by Hamilton (1943) and Yum (1931).

4.1.1.3.3. Effects of Intralist Similarity

a. Nonsense syllable lists are systematically constructed so that the items are of high

*EPAM makes stimulus and response generalization errors for the following reasons:
If a stimulus item X has already been linked to response item Y (the cue code has been established); and a similar stimulus X' is sorted in the response process; X' may be sorted to the terminal containing the image of X; the cue code for Y is found and used to make the erroneous response Y. Similarly, where the cue code for a response Y is insufficient to select out a single response from a set of items in a local part of the net, an erroneous response, Y', similar to Y, may be made.

similarity, medium similarity, and low similarity. In a typical experiment, Underwood and Richardson (1956) report:

> "The two lists with high intralist similarity had only four consonants, each being used five times. In the lists with low intralist similarity, each consonant was used only once. In each list each vowel was used twice." (pp. 119-120)

b. A major characteristic of EPAM's behavior is this: that as similarity of items in a list increases, learning effort (as measured in total learning time, or in trials to criterion) increases. This is cause for no surprise. The major task of the system is to discriminate among the items on the list; the harder we make this task (by making the items on the list more and more similar), the more effort will the system have to expend in its discrimination learning.*

---

*Specifically, an examination of the F1 process will reveal that most of the processing effort is being invested in examining positions in code and image for differences upon which differentiating tests can be constructed. As list items become more and more similar (which means that their coded forms get closer and closer to identity), it becomes increasingly more difficult for F1 to find differences (for

c. Underwood (1951, 1952, 1954, 1956) has carried out a series of studies of intralist similarity. His most consistent result is that the higher the intralist similarity, the more difficult is the learning. Measured in terms of trials-to-criterion, the ratio of effort on high similarity lists to effort on low similarity lists was approximately 1.4 to 1.

4.1.1.3.4 The Oscillation Phenomenon

a. After n trials in a rote learning task, a subject who has been making the correct response to a given stimulus item suddenly fails to make the correct response. On subsequent trials, the correct response may reappear, fail again, reappear once more, etc. We refer to this phenomenon as oscillation.*

---

high similarity means that these differences become scarce). Search time for differences increases because FO, the rote learning process, is actually doing F1 most of the time, total learning time increases with increasing similarity of items. Preliminary computer runs indicate that the process which matches code and image for differences (D5) is far and away the most expensive process in the system, consuming between 5 and 10 times as much time as any other basic process in the system.

*Hull (1935) refers to this phenomenon as an "oscillatory cycle at the threshold of recall."

b. The following two examples of oscillation appear in the simulation.*

Method	Stimulus Syllable	Response on trial: 1 2 3 4 5 6 7
Serial Anticipation	KAG	- + + + + - +
Serial Anticipation	LUK	- - + - + + +

c. Positive evidence for the existence of oscillation is provided by Hull (1935) and Hovland (1938). Hovland suggests that the explanation for oscillation may be found in the mutual interference of list items with each other. As we develop our concept of interference later in this Part (section 4.1.2), it will become clear that oscillation turns up in the behavior of EPAM for just the reason Hovland has suggested.

4.1.1.3.5 Types of Errors and their Distribution

a) Subjects can make the following kinds of errors in serial list learning:

1) failures-to-respond. (the subject says nothing).

2) remote forward associations (the subject makes a wrong response, giving a syllable farther down the list).

*The reason for this oscillation relates to the heuristic which the F2 process uses in creating the S-R link. In any given application of F2, the cue code established contains only enough information to produce the selection of the response at that moment. If the net is grown in such a way that the

3) remote backward associations (the subject makes the wrong response, giving a syllable back toward the front end of the list).

4) extraneous errors (the subject gives some response extraneous to the list).

We shall call (2) and (3), taken together, intralist errors. We shall omit all consideration of (4) because our simulation does not treat these errors, and because they do not enter into experimental studies in any consistent or important way.

b) Table 4.1 is an analysis of the errors made by EPAM in the single-list experiments. The results are as follows:

1) Intralist errors. EPAM makes remote forward association errors. It also makes remote backward association errors.*

2) Distribution of errors. The distribution of total errors is the familiar

---

response is repositioned one or more levels below its original position, then the information which was previously sufficient to retrieve the response will no longer be sufficient. In the event of insufficient information to execute tests, D2 will choose a random alternative from the last subset of images selected. The probability of a response error then becomes high.

*We are not mentioning the trivial here. There are theories of verbal learning that do not easily account for remote backward association errors (e.g., Lepley (1934) and Hull (1940))

SERIAL-ANTICIPATION (SA)

Serial Position	Failures-to-Respond	Remote Forward Associations	Remote Backward Associations	Total Intralist Errors	Total Errors
1	2	0	0	0	2
2	3	0	0	0	3
3	5	0	0	0	5
4	4	1	0	1	5
5	3	0	0	0	3
6	-	-	-	-	-
Total	17	1	0	1	18
% Total	95	5	0	5	100

PAIRED ASSOCIATES (PA)

Serial Position	Failures-to-Respond	Remote Forward Associations	Remote Backward Associations	Total Intralist Errors	Total Errors
1	1	0	0	0	1
2	4	1	1	2	6
3	3	0	1	1	4
Total	8	1	2	3	11
% Total	73	9	18	27	100

Table 4.1. An analysis of errors made by EPAM in the simulation presented in section 4.1.1.2.

bowed serial position curve. Failures-to-respond are also distributed by a bowed serial position curve.

3) Failures-to-respond. Failures-to-respond account for an overwhelming proportion of total errors under both methods.

c. Deese and Kresse (1952) have run a study, of which their Experiment I is applicable here. They found:

1) Total errors and failures-to-respond were distributed by the ordinary bowed serial position curve. Intralist errors were distributed by a serial position curve much flatter than that for total errors.

2) Failures-to-respond predominate over intralist errors by a factor of 5 to 1.*

Atkinson (1954), in a very similar study, obtained the same qualitative results, though his findings differed quantitatively somewhat from the Deese and Kresse experiment.

*The behavior of EPAM is generally consistent with Deese and Kreese's empirical results. Quantitatively, there is some disparity. The percentage they got for intralist errors/total errors for the SA experiment is about 22%, while EPAM's was 5%. However, because of the simplifications and non-precision of the EPAM simulation herein presented, and because the experiments were far from identical, one can not hope to capture such quantitative congruences at this time.

4.1.1.3.6 Serial-Anticipation Learning vs Paired Associates Learning

a) These two methods of organizing materials in a rote learning task have been discussed in section 2.2.

b) A little bookkeeping on EPAM's effort allocation reveals:

1) the same number of applications of F1 are performed in both situations. The final nets turn out to look somewhat different because of the order in which F1 is applied to the various syllables.

2) the number of S-R associations formed (F2) is 3 for PA and 5 for SA (or, in general, for the learning of n syllables, in series or in pairs, SA has n-1 associations formed, and PA has n/2.

When n is in the range of 12-18 syllables, the added number of F2 processes in the SA situation will cause a substantial increase in the ratio of learning time in SA to learning time in an equivalent PA.

c) We have not been able to locate data which bears explicitly on this point.

4.1.1.4 Discussion of Results

EPAM II is, in one sense, a theory of discrimination and generalization. It is in general agreement with two previously stated positions.

Lashley and Wade (1946) have examined the Pavlovian theory of generalization, and have framed the following hypothesis:

> "Stimulus Generalization is generalization only in the sense of failure to note distinguishing characteristics of the stimulus or to associate them with the conditioned reaction. A definite attribute of the stimulus is 'abstracted' and forms the basis of reaction; other attributes are either not sensed at all or are disregarded." (p. 81)

They call such generalization, "generalization by 'default' ", and note:

> "The stimulus generalization of conditioned reflex theory is evidently nothing more than this failure to observe and form effective associations with specific aspects of the stimulus." (p. 82)

E. J. Gibson (1940) has made a major contribution to the theory of generalization. In a limited sense, our theory can be considered to be a restatement of part of Gibson's theory, in terms of the information processing which must take place to differentiate and associate items. Gibson's thesis is essentially this: verbal items to be learned in a rote serial learning task (stimulus items and response items) will have a tendency to become confused with each other; this confusion will increase as the similarity

of items is increased; these confusions will lead to generalization errors; and these errors can be corrected only by successively differentiating the confused items from each other.* As we have seen, EPAM takes the same stand in its discrimination learning process (F1).

4.1.2 Two-List Experiments with Nonsense Syllables

In the study of rote serial learning, experiments of the following type have been run:

Learn List A
Learn List B
Retest the List A Learning

Usually, there is a decrease in the correct A responses due to the interpolated B learning, and this phenomenon has been called "retroactive inhibition."

We wish to examine the behavior of EPAM in such an experiment.

4.1.2.1 Experimental Arrangement

4.1.2.1.1 Environmental Conditions

Method: Serial Anticipation (see 2.2)

Materials: The following two lists of six nonsense syllables each:

---

*Woodworth and Schlosberg (1954) reference Gibson as follows (p. 712):

"As E. J. Gibson pointed out, (1940, 1942), 'A major necessity of verbal learning is the establishment of discrimination among the items to be learned,' or, 'Each item must become unique, discriminable from the others, in order that correct linkages can be formed.'"

LIST A	LIST B
KAG	LEK
LUK	VIL
RIL	ZAJ
PEM	PON
ROM	ZIB
TIL	QET

Criterion: one perfect run-through

4.1.2.1.2 EPAM Initial States

EPAM initial states are the same as those given in section 4.1.1.1.2.

4.1.2.2 The Simulation

In the interests of brevity, we shall not present all the details of the List B learning, for substantially no new information about the behavior of EPAM would be forthcoming from such an analysis. Instead, we present the final selection net, U2, obtained after the learning of List A followed by List B (Figure 4.1); and the actual behavior generated with this net in the List A retest trial (Table 4.2)

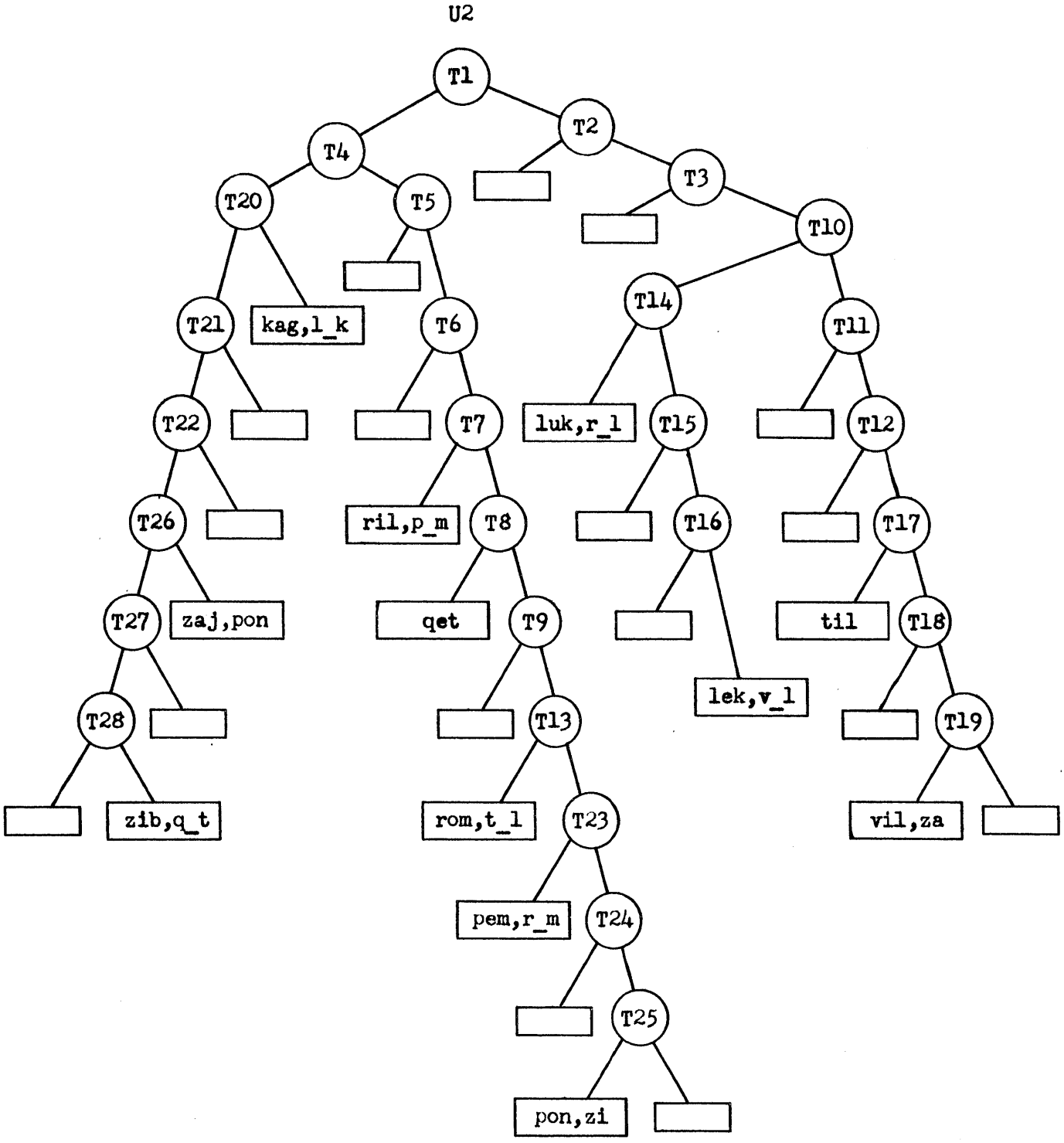
4.1.2.3 Results: Retroactive Inhibition

a) The basic phenomenon has been described above. It is generally the case that the interpolated learning (List B) "interferes" with the original learning, deteriorating performance on a retest of original learning.

b) Table 4.2 contains the stimulus-response information obtained in the retest of the List A

Text continues on page 117

Figure 4.3: Final Selection Net U2, after the processing of two lists of six nonsense syllables*



* Descriptive comments will be found on the next page.

Figure 4.3: Final Selection Net U2, after the processing of two lists of six nonsense syllables (Continued)

Comments:

a. T1, T2, T3, T4, T5, T6, T13, T17, T18, T19, T20, T21, T22, are all first letter tests.

b. T14, T15, T16, T23, T24, T25, T26, T27, T28, are second letter tests.

c. T7, T8, T9, T10, T11, T12, are third letter tests.

Table 4.2 Behavior of EPAM on the single retest of List A Syllables following the learning of List B.

Stimulus Item	Response Item	Comment
KAG	LEK*	The L_K cue code information is not sufficient to distinguish LUK from LEK. LEK was chosen by chance, by D2.
LUK	RIL	Correct response
RIL	no response*	Second-letter tests distinguish PEM from PON in the net, but second - letter information was not available in the cue code at RIL. No response was chosen by chance.
PEM	ROM	correct response
ROM	TIL	correct response

* indicates a response error

syllables following the learning of List B. The occurrance of correct responses in the retest is 60% of the criterion. If we were to carry the retest through one more trial, the inhibition effect would disappear completely as the system corrected its lack of cue-code information.

c) It is hardly necessary to cite evidence for the existence of the retroactive inhibition effect for it has been a favorite area of study for experimental psychologists over the past half century. Two good reviews of the literature exist (Britt, 1935; Swenson, 1941). Worthy of note and interest are the early studies by Muller and Pilzecker (reported in Britt (1935)); the work of DeCamp (1915) on the effect of interpolated learning; the work of Robinson (1927) on similarity and interference; and E. J. Gibson's experiment (1941) on generalization and retroactive inhibition.

Thune and Underwood (1943) report that the retroactive inhibition effect is highly transitory, disappearing in one, or a very few, trials.

4.1.2.4 Discussion

4.1.2.4.1 Interference

From the data of the simulation, we see that inter-list interference did take place. What is the operational meaning of "interference" in

terms of the EPAM system constructs? Interference occurs because in the learning of the items of List B by the F1 process, the selection net was grown to resolve certain conflicts between list A items (already learned) and list B items (in the process of being learned). Cue code information stored with List A images--information previously sufficient to retrieve all of the A responses--is in some cases no longer sufficient for this purpose. Consequently, in the response process during the retest phase, D2 must select randomly from the "ambiguous" subset, thereby producing some wrong responses. Evidence that this is not in any sense a rare situation is given by the fact that two of the five retest responses were "lost."

Thus, the model is in substantial agreement, at least qualitatively, with the classical retroactive inhibition studies.

By this theory of interference, the retroactive inhibition and oscillation phenomena are cousins. The latter is caused by items on a list interfering with each other, while the former is caused by items on an interpolated list interfering with items previously learned.

4.1.2.4.2 Forgetting

The foregoing discussion leads us to make some comments about the nature of forgetting. Forgetting is generally thought of to be the result of the decay or destruction of information in the memory of a subject. Certainly this is a simple hypothesis, with much appeal, and it may be true. Our work suggests to us an alternative, which is this: forgetting is not the result of the loss of information but of the more or less temporary mislocation of information in a memory net structure, which tends to interrupt, temporarily, S-R links. To be sure, retrieval of the "forgotten" information is possible again by adding more cues to cue codes. But the system has no way of "checking back" to see what S-R links have been interrupted when its net is grown, and can only take corrective action when R is called for in response to S. If such a relearning opportunity is not available soon after original learning (before much interpolated learning takes place), learned material may become more or less "permanently" lost, retrievable only when sufficient cues have been amassed. There is some inconclusive evidence

from neuro-physiological studies which indicates that information in human brains may never really be destroyed over time. (Penfield, 1959) If this is the case, then at least our model provides a mechanism for explaining the forgetting phenomenon in the absence of actual information loss.

4.2 SUMMARY

EPAM is a program for an IPL-V computer which realizes the theoretical model of human rote learning behavior presented in Part 2. In this Part, the behavior of EPAM in a number of typical rote learning tasks was examined. The results are summed up as follows:

1) if n is the number of alternative stimuli, EPAM's disjunctive reaction time is approximately equal to $(a + b \log_2 n)$.
2) EPAM makes errors of stimulus generalization and response generalization.
3) EPAM expends considerably more effort in learning lists with high similarity among items than in learning lists with low similarity among items.
4) Examples of oscillation of responses were present in the behavior.
5) Failures-to-respond predominated over intrusion errors. Total errors and failures-to-respond were distributed

by the ordinary serial position curve. Intrusion errors included remote forward and remote backward associations.

6) EPAM expends more effort in learning a list of n syllables by the serial-anticipation method than it does in learning the equivalent list of associate pairs (n/2 pairs, using the same syllables)

7) The learning of two lists of items produced inter-list interference between the list A items and the list B items, resulting in retroactive inhibition--the degradation of system performance in the retest of list A items.

In general, these results are in quite good qualitative agreement with empirical evidence, thus giving some basis for the credibility of the theory.

CONCLUDING REMARKS

In Part 4, we presented some general consequences of the EPAM model in standard rote learning situations, and these were seen to agree reasonably well with available data on human behavior in these situations. EPAM represents an initial attempt at framing an information processing theory of human verbal learning. It is not the last word. Hull has expressed well what we feel:

> "The history of science shows that scientific theory advances by a series of successive approximations, that it is to a certain extent a true trial-and-error process. The indicators of error are primarily failures of the theorems of the system to agree with relevant facts. In general, each successive trial eliminates some of the evidences of error contained in preceding attempts, it extends the system to include a wider range of known fact and, perhaps most important of all, it projects its deductions into new regions where observations have not yet been made. There is no reason to believe that the evolution of theory in the behavioral (social) sciences will be exceptional in this respect."*

We would like to summarize where we have been and what we still have to do. EPAM was conceived as a serial information processor, capable of doing only one thing at a time, accomplishing various learning tasks by time-sharing among some basic processes for comparing and moving and storing information symbols. An attempt to define these processes produced many "special purpose" mechanisms to account for various rote learning phenomena. Gradually, these special purpose "devices" yielded to general purpose schemes for discriminating stimuli from each

---

*Hull, et. al. (1940), p. 305.

other and for associating these stimuli together. Thus, there came into being the discrimination net, the concept of an internal image, and the concept of the cue code. The basic ideas were given a precise statement as programs for a digital computer. The resulting system of information processes, as we have said, exhibited interesting behavioral characteristics, some of which we had not anticipated.

Where do we take this research from here?

A. Empirical Exploration of the Model. In Part 4 we could do little more than point out some of the general features of the behavior of EPAM. IPL-V will soon exist, and we will be able to draw out the implications of EPAM in great detail. This phase will consist of putting EPAM through a wide variety of verbal learning experiments for which data on human learning is available. The behavior of the theoretical system will be closely examined and the sufficiencies and deficiencies noted and explored. A brief and tentative rundown of experiments which we hope to reproduce is the following:

1) A study of remote forward and backward associations (McGeoch, 1936); of anticipation errors (Lumley, 1932; Hull, 1935)

2) Forward vs backward serial learning (Ribback and Underwood, 1950)

3) Oscillation as a function of serial position (Hovland, 1938)

4) Transfer studies involving the manipulation of stimulus similarity and response similarity (Yum, 1931; Bruce, 1933; Gibson, 1941, 1942; Hamilton, 1943)

5) Familiarization training study by Hovland and Kurtz (1952)

6) Retroactive Inhibition as a function of intralist and interlist similarity. (Gibson, 1941)

B. Modifications of the Theory. In the exploration process just described, we are certain to find important incompletenesses in the model. We are also bound to discover results which are qualitatively, but not quantitatively, accurate. We will be forced to modify and extend our model. Examples are already in evidence:

1) With respect to images, we do not yet have a clear idea of just how much information is in an internal image "sufficient" for efficient processing, or of precisely how to get the system to regulate this level of sufficiency by itself. Currently, images contain <u>all</u> of the information available in the external code. We get the feeling, in

simulating the model, that the system "knows too much," that though it makes errors, it makes too few errors.

2) With respect to failures-to-respond, exploration will be needed to determine just how much redundancy the system will systematically have to introduce into its selection net to produce the correct proportion of response failures to intrusion errors.

3) As an example of a phenomenon which the system is not capable of explaining, we cite the phenomenon of proactive inhibition. This effect derives from mechanisms beyond those proposed in the model. In this particular case, we do have a clear idea of how to proceed to extend our model in a very general way to account for the phenomenon (it involves the addition of a third kind of memory to our system). This idea will have to be well formulated, developed precisely as a program, and then tested in conjunction with the existing processes.

C. Predictions and Testing. In conjunction with the exploration and modification of the model, specific predictions about hitherto unsearched for behavior in particular kinds of experiments will be generated.

These predictions will be used as one basis for testing the various postulates of the system.

From the work already done with EPAM I, we find that it is possible to make very precise and meaningful predictions of behavior, and that these predictions afford substantial grounds for the acceptance or rejection of an hypothesis. This has given us confidence and high hopes that eventually, by a rigorous procedure of formulation and testing, we may be able to arrive at a complete theory of verbal learning.

BIBLIOGRAPHY

Atkinson, R. C., "An Analysis of Rote Serial Learning in Terms of a Statistical Model," Indiana University, Unpublished Doctoral Dissertation, 1954.

Bricker, P. D., "Information Measurement and Reaction Time: A Review," in Quastler, H. (Ed.). Information Theory in Psychology, Glencoe: Free Press, 1955.

Britt, S. H., Retroactive Inhibition: A Review of the Literature, Psychol. Bull., Vol. 32, 1935.

Bruce, R. W., Conditions of Transfer of Training, J. Exp. Psychol., Vol. 16, 1933, pp. 343-361.

Crossman, E. R., Entropy and Choice Time: The Effect of Frequency Unbalance on Choice-Response, Quart. J. Exp. Psychol., Vol. 5, 1953, pp. 41-51.

DeCamp, J. E., A Study of Retroactive Inhibition, Psychol. Monogr., Vol. 19, No. 4, 1915.

Deese, J. and F. H. Kresse, An Experimental Analysis of the Errors in Rote Serial Learning, J. Exp. Psychol., Vol. 44, 1952, pp. 199-202.

Dinneen, G. P., Programming Pattern Recognition, Proceedings of the Western Joint Computer Conference, March, 1955.

Ebbinghaus, H., Memory, (Translated by H. A. Ruger and C. E. Bussenius, New York: Teachers College, 1913.) 1885.

Feigenbaum, E. A., and H. A. Simon, "A Theory of the Serial Position Effect," CIP Working Paper #14, Carnegie Institute of Technology (Graduate School of Industrial Administration), Pittsburgh, Pa., 1959.

Feldman, J., An Analysis of Behavior in Two Choice Situations, Carnegie Institute of Technology, Unpublished Doctoral Dissertation, 1959.

Finkenbinder, E. O., The Curve of Forgetting, Amer. J. Psychol., Vol. 24, 1913.

Gelernter, H. L. and N. Rochester, Intelligent Behavior in Problem-Solving Machines, IBM Journal of Research and Development, Vol. 2, 4, October, 1958.

Gibson, E. J., A Systematic Application of the Concepts of Generalization and Differentiation to Verbal Learning, Psychol. Rev., Vol. 47, 1940, pp. 196-229.

_____, Retroactive Inhibition as a Function of Degree of Generalization Between Tasks, J. Exp. Psychol., Vol. 28, 1941, pp. 93-115.

_____, Intra-List Generalization as a Factor in Verbal Learning, J. Exp. Psychol., Vol. 30, 1942, pp. 185-200.

Goldstine, H. H. and J. von Neumann, Planning and Coding of Problems for an Electronic Computing Instrument, Part 2, Volume 1, Princeton: Institute for Advanced Study, 1947.

_____, ibid, Part 2, Volume 2, 1948.

_____, ibid, Part 2, Volume 3, 1948.

Hamilton, R. J., Retroactive Facilitation as a Function of Degree of Generalization Between Tasks, J. Exp. Psychol. Vol. 32, 1943, pp. 363-376.

Hebb, D. O., The Organization of Behavior: A Neuropsychological Theory, New York: Wiley, 1949.

Hick, W. E., On the Rate of Gain of Information, Quart. J. Exp. Psychol., Vol. 4, 1952, pp. 11-26.

Hovland, C. I., Experimental Studies in Rote Learning Theory, III, Distribution of Practice with Varying Speeds of Syllable Presentation, J. Exp. Psychol., Vol. 23, 1938, pp. 172-190.

Hovland, C. I. and Kurtz, D. H., Experimental Studies in Rote Learning Theory, X, Prelearning Syllable Familiarization and the Length-Difficulty Relationship, Vol. 44, 1952, pp. 31-39.

Hull, C. L., The Influence of Caffeine and Other Factors on Certain Phenomena of Rote Learning, J. Gen. Psychol., Vol. 13, 1935, pp. 249-273.

_____, The Conflicting Psychologies of Learning--A Way Out, Psych. Rev., Vol. 42, 1935, pp. 491-516.

Hull, C. L., Hovland, C. I., Ross, R. T., Hall, M., Perkins, D. T. and F. B. Fitch, Mathematico-deductive Theory of Rote Learning, New Haven: Yale University Press, 1940.

Hyman, R., Stimulus Information as a Determinant of Reaction Time, J. Exp. Psychol., Vol. 45, 1953, pp. 188-196.

James, W., The Principles of Psychology, New York: Holt, 1890.

Lashley, K. S., and Wade, M., The Pavlovian Theory of Generalization, Psychological Review, Vol. 53, 1946, pp. 72-87.

Lepley, W. M., Serial Reactions Considered as Conditioned Responses, Psychol. Monogr., Vol. 46, 1934, No. 205.

Lumley, F. H., Anticipation of Correct Responses as a Source of Error in the Learning of Serial Responses, J. Exp. Psychol., Vol. 15, 1932, pp. 195-205.

McCrary, J. W., and W. S. Hunter, "Serial Position Curves in Verbal Learning," Science, Vol. 117, 1953.

McGeoch, J. A., The Influence of Degree of Learning upon Retroactive Inhibition, Amer. J. Psychol., Vol. 41, 1959, pp. 252-262.

_____, Studies in Retroactive Inhibition, II, Relationships Between Temporal Point of Interpolation, Length of Interval, and Amount of Retroactive Inhibition, J. Gener. Psychol., Vol. 9, 1933, pp. 44-57.

_____, The Direction and Extent of Intra-serial Associations at Recall, Amer. J. Psychol., Vol. 48, 1936, pp. 221-245.

McGeoch, J. A. and A. L. Irion, The Psychology of Human Learning, New York: Longmans, 1952.

Mowbray, G. H. and M. U. Rhoades, On the Reduction of Choice Reaction Times with Practive, Quar. J. of Exp. Psychol., Vol. XI, No. 1, February, 1959.

Newell, A. and H. A. Simon, The Logic Theory Machine, Trans. on Information Theory, Vol. IT-2, No. 3, September, 1956.

Newell, A., Shaw, J. C. and H. A. Simon, Empirical Explorations of the Logic Theory Machine, Proceedings of the Western Joint Computer Conference, IRE, February, 1957.

_____, The Elements of a Theory of Human Problem Solving, Psych. Rev., Vol. 63, March, 1958.

_____, The Processes of Creative Thinking, The RAND Corporation Paper, P-1320, August, 1958.

_____, Chess Playing Programs and the Problem of Complexity, IBM Journal of Research and Development, Vol. 2, 4, October, 1958.

_____, Report on a General Problem Solving Program, The RAND Corporation Paper, P-1584, January, 1959.

_____, The Microanalysis of Human Problem Solving Behavior, paper given before the Eastern Psychological Association meetings, April 4, 1959.

Newell, A., Tonge, F., Feigenbaum, E., Mealy, G. and N. Saber, Manual for Information Processing Language V, CIP Working paper #16, Carnegie Institute of Technology (Graduate School of Industrial Administration), Pittsburgh, Pa., 1959.

Penfield, W., The Interpretive Cortex, Science, Vol. 129, No. 3365, June 26, 1959, pp. 1719-1725.

Perlis, A., Smith, J., and VanZoren, H., The IT Language for the IBM 650, Carnegie Institute of Technology, 1957.

Ribback, A. and B. J. Underwood, An Empirical Explanation of the Skewness of the Bowed Serial Position Curve, J. Exp. Psychol., Vol. 40, 1950, pp. 329-335.

Robinson, E. S., The 'Similarity' Factor in Retroaction, Amer. J. Psychol., Vol. 39, 1927, pp. 297-312.

Rosenblatt, F., The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain, Psych. Rev., Vol. 6, No. 65, November, 1958.

Selfridge, O. G., Pattern Recognition and Modern Computers, Proceedings of the 1955 Western Joint Computer Conference, IRE, March, 1955.

Simon, H. A., Models of Man, New York: Wiley, 1957.

Stevens, S. S., ed., Handbook of Experimental Psychology, New York: Wiley, 1951

Swenson, E. J., Retroactive Inhibition: A Review of the Literature, Univ. Minn. Stud. Educ., Vol. 1, 1941.

Thune, L. E. and B. J. Underwood, Retroactive Inhibition as a function of Degree of Interpolated Learning, J. Exp. Psychol., Vol. 32, 1943, pp. 185-200.

Tonge, F., Development of a Heuristic Program for an Assembly Line Balancing Problem, CIP Working Paper #10, Carnegie Institute of Technology (Graduate School of Industrial Administration), Pittsburgh, Pa., 1958.

Underwood, B. J., Studies of Distributed Practice: VII. Learning and Retention of Serial Nonsense Lists as a Function of Intralist Similarity, J. Exp. Psychol., Vol. 42, No. 2, 1952, pp. 80-89.

_____, Intralist Similarity in Verbal Learning and Retention, Psych. Rev., Vol. 61, No. 3, 1954, pp. 160-166.

Underwood, B. J. and D. Good, Studies of Distributed Practice: I. The Influence of Intra-List Similarity in Serial Learning, J. Exp. Psychol., Vol. 42, No. 2, 1951, pp. 125-133.

Underwood, B. J. and J. Richardson, The Influence of Meaningfulness, Intralist Similarity, and Serial Position on Retention, J. Exp. Psychol., Vol. 52, No. 2, 1956, pp. 119-126.

Woodworth, R. S. and H. Schlosberg, Experimental Psychology, New York: Holt, 1956.

Yum, K. S., An Experimental Test of the Law of Assimilation, J. Exp. Psychol., Vol. 14, 1931, pp. 68-82.

APPENDIX

THE COMPLETE CODE FOR EPAM II IN IPL V, INCLUDING PROGRAMS AND DATA LIST STRUCTURES.

PROGRAMS AND LIST STRUCTURES ARE ARRANGED ALPHABETICALLY, ACCORDING TO THE IDENTIFICATION AND SEQUENCE NUMBERS ON THE RIGHT SIDE OF THE PAGE. NO SPECIFIC DATA INPUTS ARE SHOWN, WITH THE EXCEPTION OF AN EXAMPLE OF A CAPITAL LETTER CODE, GIVEN AS Q01.

```
                A0      10A0     J10                A0000
                A1      10A1     J10                A0001
                A2      10A2     J10                A0002
                A3      10A3     J10                A0003
                A4      10A4     J10                A0004
                A5      10A5     J10                A0005
                A6      10A6     J10                A0006
                A7      10A7     J10                A0007
                A8      10A8     J10                A0008
                L    1  21    A                     CAPL01
                L    2  21    B                     CAPL02
                L    3  21    C                     CAPL03
                L    4  21    D                     CAPL04
                L    5  21    E                     CAPL05
                L    6  21    F                     CAPL06
                L    7  21    G                     CAPL07
                L    8  21    H                     CAPL08
                L    9  21    I                     CAPL09
                L   10  21    J                     CAPL10
                L   11  21    K                     CAPL11
                L   12  21    L                     CAPL12
                L   13  21    M                     CAPL13
                L   14  21    N                     CAPL14
                L   15  21    O                     CAPL15
                L   16  21    P                     CAPL16
                L   17  21    Q                     CAPL17
                L   18  21    R                     CAPL18
                L   19  21    S                     CAPL19
                L   20  21    T                     CAPL20
                L   21  21    U                     CAPL21
                L   22  21    V                     CAPL22
                L   23  21    W                     CAPL23
                L   24  21    X                     CAPL24
                L   25  21    Y                     CAPL25
ALPHABET        L   26  21    Z                     CAPL26
FORMAT OF A DIFFERENCE
              1 9   0     0000                      DIFF 1
INTEGER DESIGNATING LINE
              1           N3                        DIFF 2
INTEGER DESIGNATING BIT
              1           N2                        DIFF 3
SIGN OF DIFFERENCE.
              1           J3      0000              DIFF 4
SIMILARLY,
              1                                     DIFF 5
              1 9   1     0000                      DIFF 6
              1           N   1                     DIFF 7
              1           N  10                     DIFF 8
              1           J   4   0000              DIFF 9
DO,NET INTERPRETER, (0) IS NAME
                DO        J42                       D00 01
OF NET OUTPUT (0) IS NAME OF
                        20W   2       SAVE NAME     D00 02
RESPONSE LIST
                          J  90                     D00 03
                        60Y   0                     D00 04
```

```
                                    11W   2                          D00 05
                                    10A   3                          D00 06
IDENTIFY ORIGIN OF TRACKING           J11    93                       D00 07
                                9   3 11W   2                        D00 08
                                    10A   0                          D00 09
                                      J  10                          D00 10
95 IS ERROR STOP                    70J   7                          D00 11
                                      J   1                          D00 12
                                    709   1                          D00 13
                                    11W   2                          D00 14
                                      J  81                          D00 15
                                      J   1                          D00 16
                                    11Y   0                          D00 17
ADDS TO TRACK                         E   1                          D00 18
BRANCH RIGHT ON H5 +                70        9   2                  D00 19
                                    11W   2                          D00 20
                                      J  82 9   4                    D00 21
                                9   2 11W   2                        D00 22
                                      J  83 9   4                    D00 23
                                94    11W2                           D00 240
                                    20Y5                             D00 241
                                    20W2     93                      D00 242
                                9   1 11W   2                        D00 25
SAVE NAME OF TERMINAL               60Y4                             D00 251
                                    10A   1                          D00 26
FIND DICTIONARY ENTRY                 J  10                          D00 27
ERROR STOP                          70J   7                          D00 28
GETS NAME OF RESPONSE LIST          12H   0                          D00 29
                                      J   6                          D00 30
                                      J  10                          D00 31
                                    70J   7                          D00 32
                                    60W   1                          D00 33
                                    11Y   0                          D00 34
                                    10A   2                          D00 35
ASSIGN TRACKING LIST TO RESPONSE      J  11                          D00 36
                                    11W   1 J  32                    D00 37
WRITER. D1. TRANSFORMS A LIST   1                                    D01 01
OF LETTER IMAGES TO THE         1                                    D01 02
PRINTED WORD. IMAGES ARE IN     1                                    D01 03
THE READ-WRITE MODE. INPUT      1                                    D01 04
(0) IS AN EPAM IMAGE.           1                                    D01 05
                              D1      92                             D01 06
                                    70J152                           D01
                                      J41                            D01 061
                                    40Y3                             D01 061
                                    10X0                             D01 062
                                    20Y3                             D01 063
```

```
                                        J90                 D01
                                      20W0                  D01
                                      1090                  D01 07
GENERATE LETTERS OF IMAGE                J100               D01 08
1W0 IS LIST OF DATA TERMS             11W0                  D01 09
PRINT LIST OF DATA TERMS                 E4                 D01 10
                                      30Y3                  D01 101
                                      11W0                  D01 11
ERASE LIST                               J71    J31         D01 12
                                  90     91                  D01
                                      10U1                  D01 16
                                      11W1                  D01 17
                                        D2                  D01 18
                                      11W0                  D01 19
                                        J6                  D01 20
                                        J65                 D01 21
                                      11W1                  D01 22
                                        J71    J4          D01 23
CREATE LETTER ITEM IN W1              91     04J90          D01 24
USING IMAGE (0).                             60W1          D01 25
                                        J6     J64         D01 26
                                      92     40H0   D12     D01
D2, SORT ITEM FOR DISCRIMINATION      1                     D0201
INPUT (0)=ITEM, (1) = NET             1                     D0202
OUTPUT (0)=NAME OF IMAGE LIST         1                     D0203
AT END OF PROCESS, 1Y4 IS NAME        1                     D0204
OF TERMINAL CELL, 1Y5 IS NAME OF      1                     D0205
NODE POINTING AT THIS TERMINAL CELL   1                     D0206
                                      D2     20Y1   D0      D0207
D5, PRODUCE A LIST OF DIFFERENCES     1                     D0501
BETWEEN ITEMS(0)THE INCUMBENT, AND    1                     D0502
(1), THE CHALLENGER NUMBER OF         1                     D0503
DIFFERENCES FOUND IS CONTROLLED       1                     D0504
BY PARAMETER P2. ALSO REORDERS        1                     D0505
NOTICING LISTS, BASED ON OUTCOME      1                     D0506
                                      D5       J43          D0507
                                               J21          D0508
                                             10N0           D0509
CREATE ZERO NUMBER FOR COUNT                   J120         D0510
                                             20W3           D0511
                                               J90          D0512
CREATE LIST FOR DIFFERENCES                  20W2           D0513
9-0 IS SCANNER FOR D18                       1090           D0514
NOTICING GENERATOR                             D18          D0515
ERASE COUNT CELL                             11W3           D0516
                                               J9           D0517
                                             11W2           D0518
```

```
UPDATES NOTICING LISTS, USING DIFFE                1091                    D0519
                                                     J100                  D0520
                                                   11W2     J33            D0521
9-0 NOTICES DIFFERENCES                    90      20K0          SAVE PAIR D0522
                                                   60K1                    D0523
                                                   11K0                    D0524
INPUT THE CHALLENGER                               11W1                    D0525
FINDS I,JTH BIT OF CHALLENGER                        E9                    D0526
ON NO FIND, SIGNAL GEN. CONTINUE                   70J4                    D0527
SAVE CHALLENGERS BIT                               60K2                    D0528
                                                   11K1                    D0529
                                                   11K0                    D0530
                                                   11W0                    D0531
FINDS I,JTH BIT OF INCUMBENT                         E9                    D0532
                                                   70J4                    D0533
MATCH BITS                                           J2                    D0534
IF NO DIFF, TO GEN. FOR CONTINUE                   70        0000          D0535
IF DIFF.,                                          11K0.                   D0536
                                                   11K1                    D0537
                                                   11K2                    D0538
BUILD THE DIFFERENCE                                 J93                   D0539
                                                   11W2                    D0540
                                                     J6                    D0541
ADD IT TO LIST OF DIFFERENCES                        J65                   D0542
                                                   11W3                    D0543
ADD TO COUNT OF DIFF.                                J125                  D0544
                                                   11P2                    D0545
                                                     J114    J5            D0546
UPDATES P0 AND P1 BASED ON                 91      40H0                    D0548
DIFFERENCES (PRESENTED BY GEN.)                      J81           LINEINDEX D0549
                                                   10P0                    D0550
LOCATES CELL BEFORE SYMB(1) ON(0)                    E10                   D0551
ER. STP.                                           70J7                    D0552
MOVES SYMBOL UP ON LIST                              E11                   D0553
                                                     J82           CODEINDEX D0554
                                                   10P1                    D0555
                                                     E10                   D0556
                                                   70J7                    D0557
CONTINUE IN GENERATOR.                               E11     J4            D0558
D7, FIND CUR LIST OF IMAGE                 1                               D0701
LIST (0).                                  1                               D0702
                                           D7      10A8    J10             D0703
D8, ADD A LETTER OF (0) TO P.R.            1                               D0801
AT (1).                                    1                               D0802
                                           D8        J44                   D0803
                                                     J21                   D0804
                                                   11W1                    D0805
```

```
                                    A8                                  D0806
                                  7091                                  D0807
                                  60W2                                  D0808
                                    J78                                 D0809
                                  7091      95                          D0810
                          95      10P0                                  D0811
                                  1092                                  D0812
                                    J100    J34                         D0813
SAVE POSITION             92      60W3                                  D0814
                                  11W2                                  D0815
                                    J6                                  D0816
                                    E3                                  D0817
                                  70J7              E.S.                D0818
SAVE LOC. IN W4                   60W4                                  D0819
                                  52H0                                  D0820
IS SYMBOL BLANK                   10K11                                 D0821
                                    J2                                  D0822
                                  70J4                                  D0823
                                  11W0                                  D0824
                                  11W3                                  D0825
                                    E3                                  D0826
                                  52H0                                  D0827
                                    J74                                 D0828
                                    J136                                D0829
                                  21W4      J3                          D0830
                          91        E6                                  D0831
STORE LIST NAME                   20W4                                  D0832
                                  10P0                                  D0833
STORE POINTER                     20W3      94                          D0834
                          94        J193                                D0835
                                  7093                                  D0836
                                  11W4                                  D0837
                                  10K11                                 D0838
                                    J64     94                          D0839
                          93      11W1                                  D0840
                                  11W4                                  D0841
                                  60W2                                  D08411
                                  10A8                                  D0842
                                    J11     95                          D0843

D9, CREATE STRUCTURE OF TESTS       1                                   D0901
ON DIFFERENCES (0), ERASING         1                                   D0902
DIFFERENCES. INPUT (1) IS           1                                   D0903
NAME OF TERMINAL AT WHICH           1                                   D0904
CONFUSION OCCURREDTHIS IS INSERTED  1                                   D0905
INTO TEST STRUCTURE AFTER FIRST     1                                   D0906
TEST OUTPUT (0) IS NAME OF TEST     1                                   D0907
STRUCTURE CREATED                   1                                   D0908
```

```
1W0=NAME OF LIST OF DIFFERENCES      D9      J44                    D0909
1W1=NAME OF CONFUSED TERMINAL                J21                    D0910
CREATE A NULL TERMINAL CELL                  90                     D0911
1W3=NAME OF NODE OR TERM PASSED UP          20W3                    D0912
                                            11W0                    D0912
                                            1091                    D0913
GENERATE DIFFERENCES FOR 9-1                 J100                   D0914
                                            11W3                    D0915
LOCATE MINUS BRANCH OF FIRST TEST            E12                    D0916
                                            60K6                    D0917
                                            52H0                    D0918
ERASE TERMINAL THERE                         J72                    D0919
                                            11W1                    D0920
INSERT CONFUSION TERMINAL                   21K6                    D0921
                                            11W0                    D0922
                                             J72                    D0923
ARRANGE OUTPUT AND QUIT                     11W3      J34           D0924
CREATE A NULL TERMINAL               90      J90                    D0925
                                            60K0                    D0926
                                            10J3                    D0927
                                            10A0                    D0928
                                             J11                    D0929
                                            11K0                    D0930
GET A NEW DICTIONARY ENTRY (X)               E8                     D0931
                                            60K1                    D0932
                                            10A1                    D0933
                                             J11                    D0934
                                            11Y3                    D0935
NULL RESPONSE                               10K10                   D0936
                                            11K1                    D0937
ENTER NULL RESP. INTO DICTIONARY             J11                    D0938
                                            11Y3                    D0939
HOUSEKEEPING                                21K1                    D0940
                                            11K0       0000         D0941
SUBPROCESS FOR GENERATOR, SETS    1                                 D0942
UP A NODE, BASED ON DIFFERENCE    1                                 D0943
SAVE NAME OF DIFFERENCE              91     60W4                    D0944
USING DIFF.,INPUT APPROPRIATE TEST           92                     D0945
CREATE NULL TERMINAL FOR-BRANCH              90                     D0946
INPUT NODE PASSED UP AS POS. BR.            11W3                    D0947
CREATE NODE                                  J93                    D0948
                                             J136                   D0948
PREPARE FOR PASSING IT UPWARDS              60W3                    D0949
MARK DESC. LIST OF NODE, CONTINUE            93      J4            D0950
                                     93     10J4                    D0951
                                            10A0                    D0952
                                             J11                    D0953
```

```
                                    11W3                          D0954
INPUT I FROM DIFFERENCE                                           D0955
                                    10A4                          D0956
                                      J11                         D0957
                                    11W3                          D0958
                                    0095                          D0959
                                    10A5        J11               D09591
                              94    11W4        J81               D0960
                              95    11W4        J82               D0961
INPUTS APPROPRIATE TEST       92    11W4                          D0962
INPUT SIGN OF DIFFERENCE            J83                           D0963
                                    J1                            D0964
                                    7096                          D0965
                                    10Z0        0000              D0966
                              96    10Z1        0000              D0967
D10, ADD TEST STRUCTURE (0) TO   1                                D1001
TESTING POINT (1) IN PLACE OF    1                                D1002
TERMINAL OF CONFUSION (2).       1                                D1003
                            D10       J52                       D1004
                                      J191                      D1005
LOCATE MINUS BRANCH                   J191                      D1006
                                    12W1                        D1007
                                    11W2                        D1008
IS THIS THE CONFUSED TERMINAL         J2                        D1009
                                    70          90              D1010
                                      J191      90              D1011
                              90    11W0                        D1012
                                    21W1        J32             D1013
D12, TEST IS (0) THE NAME        1                              D1201
OF AN IMAGE LIST. SET H5 + ON    1                              D1202
I.L., - ON NULL I.L.             1                              D1203
K10 IS THE NULL IMAGE LIST          D12   10K10                 D1204
                                      J2        J5              D1205
D15, MATCH ITEM (0) WITH ITEM    1                              D1501
(1) FOR IDENTITY. NO OUTPUT.     1                              D1502
SET H5 + ON IDENTITY, - ON       1                              D1503
NO MATCH                         1                              D1504
                                  D15   J51       90            D1505
ADVANCE POINTERS AT HIGHEST LEVEL 90      98                    D1506
                                    7091                        D1507
                                      97        93              D1508
ADVANCE POINTERS AT LEVELOF LETTERS 93    98                    D1509
                                    7094                        D1510
                                      97        92              D1511
ADVANCE POINTERS AT LEVEL OF CODE 92      98                    D1512
                                    7096                        D1513
                                    12W1                        D1514
```

```
                                            12W0                D1515
MATCH CODE SYMBOLS                            J2                D1516
                                            7095     92         D1517
                                       94     J31    90         D1518
                                       95     J31               D1519
                                              J31    J31        D1520
                                       96     J31    93         D1521
                                       91     J4     J31        D1522
                                       97   12W1                D1523
                                            12W0     J51        D1524
                                       98     J190              D1525
                                            70 0000  J191       D1526
D17, CREATE IMAGE OF ITEM (0)        1                          D1700
                                       D17    J74    J136       D1701
D18, NOTICING GENERATOR, GENERATES   1                          D1801
PAIRS, I, J, FROM NOTICING LISTS     1                          D1802
P0 (FOR LINES) AND P1 (FOR CODE).    1                          D1803
OUTPUT, I=(0), J=(1). INPUT (0)=     1                          D1804
NAME OF THE SUBPROCESS.              1                          D1805
                                       D18  10W1                D1806
                                              J17               D1807
                                            10P0                D1808
                                            20W      90         D1809
                                       90   10P1                D1810
                                            20W1                D1811
                                              J190              D1812
                                            70J19    93         D1813
                                       93     J191              D1814
                                            7090                D1815
                                            12W1                D1816
                                            12W0                D1817
                                              J18               D1818
                                            70J19    93         D1819
RESPONDER. D21. INPUT (0)=           1                          D21 01
STIMULUS ITEM. OUTPUT (0)=           1                          D21 02
RESPONSE ITEM SELECTED. CAN          1                          D21 03
BE NO RESPONSE, K10. INPUT (1)       1                          D21 04
IS THE NAME OF NET IN WHICH          1                          D21 05
RESPONSE IS TO BE MADE.              1                          D21 06
                                       D21    J51               D21 07
                                            11W1                D21 08
                                            11W0                D21 09
                                              D2                D21 10
                                              D7                D21 11
                                            7090                D21 12
                                            11W1                D21 13
                                              J6                D21 14
```

```
                                        D2      J31                 D21 15
                                90      10K10   J31                 D21 16
D22.  RECOGNIZES AND WRITES       1                                 D22 01
A STIMULUS (0) FROM A NET (1)     1                                 D22 02
                                D22     D2                          D22 03
                                        40H0                        D22 04
                                        10K10                       D22 05
                                          J2                        D22 06
                                        70D1                        D22 07
                                          J152          0           D22 08
D23.  TEST FOR RECOGNITION OF     1                                 D23 01
ITEM.  H5 + ON RECOGNIZING, - ON  1                                 D23 02
NON-RECOGNITION.  INPUT (0) = NAME 1                                D23 03
OF ITEM.  (1) = NAME OF NET       1                                 D23 04
SAVE ITEM                       D23     6090                        D23 05
SORT ITEM                               D2                          D23 06
                                        40H0                        D23 07
                                        10K10                       D23 08
IS IMAGE NULL                             J2                        D23 09
                                        70          91              D23 10
                                        1190        D15             D23 11
                                90                0         0       D23 12
                                91      30H0        J3              D23 13
D24, PRODUCE ANTICIPATION TO (0), 1                                 D24 01
LEAVING ANTICIPATION AS OUTPUT,   1                                 D24 02
AND PRINTING IT.  NAME OF NET     1                                 D24 03
IS INPUT (1).                     1                                 D24 04
                                D24     J51                         D24 05
                                        11W1                        D24 06
                                        11W0                        D24 07
MAKE RESPONSE INTERNALLY                  D21                       D24 08
                                        40H0                        D24 09
                                          D1        J31             D24 10
D27, SUBJECT CHECKS HIS           D27     J51                       D27 01
RESPONSE.  INPUT (0) AND (1), TWO       11W1                        D27 02
ITEMS TO BE CHECKED.                    11W0                        D27 03
                                          D15                       D27 04
                                        70J31                     D27 05
                                        11W1                        D27 06
                                        10K10                       D27 07
                                          J2                        D27 08
                                        7090                        D27 09
                                        11W0                        D27 10
                                        10K10                       D27 11
                                          J2        J31             D27 111
                                90      11W0                        D27 12
                                        10K10                       D27 13
```

```
                                                       J2                                 D27 14
                                                       J5      J31                        D27 141
E3, LOCATES SYMBOL N= (0) ON                 E    1 11H     5 J   65                      E01 01
LIST (1). OUTPUT IS LOCATED                  E    3   J    51                             E03 01
CELL.  SET H5 - ON NO LOCATE                        10S     0                             E03 02
                                                      J124    92                          E03 03
                                             92       J191                                E03 04
                                                    709     1                             E03 06
                                                      J 125                               E03 08
                                                    40H     0                             E03 09
                                                    11W    .0                             E03 10
                                                      J 114                               E03 11
                                                    709     2                             E03 12
                                                    30H     0                             E03 13
                                                    11W     1 J   31                      E03 14
                                             9    1 30H     0 J   31                      E03 15
E4, PRINTS LIST (0) OF DATA              1                                                E04 01
TERMS WITHOUT NAMES                      1                                                E04 02
                                             E    4 40H     0                             E04 03
                                                      J152    91                          E04 04
                                             9    1   J    60                             E04 05
                                                    70J     8                             E04 06
                                                    12H     0                             E04 07
                                                      J 153 9     1                       E04 08
                                             E    5 109     0 J   16                      E05 01
                                             9    0   9     1  0000                       E05 02
                                             9    1     0000                              E05 03
                                                      J     3                             E05 04
                                                      N     5                             E05 05
                                                      J     4                             E05 06
                                                      N     5  0000                       E05 07
E6, CREATE A LOCAL SYMBOL                    E6       J90     J136                        E0601
E7, CREATE REGIONAL SYMBOL OF                E7       J90                                 E0701
TYPE (0)                                              J6      J135                        E0702
E8, CREATE SYMBOL IN X REGION                E8     10X0      E7                          E0801
E9, FINDS JTH BIT OF LINE I OF           1                                                E0901
AN ITEM. ITEM=(0), I=(1), J=(2).         1                                                E0902
OUTPUT (0) = SYMBOL FOUND. SET H5+       1                                                E0903
ON FIND,-ON NO FIND.                     1                                                E0904
                                             E9       J52                                 E0906
                                                    11W0                                  E0907
                                                    11W1                                  E0908
LOCATES ITH LINE                                      E3                                  E0909
                                                    70J32                                 E0910
                                                    52H0                                  E0911
                                                      J82                                 E0912
                                                    11W2                                  E0913
```

```
LOCATES JTH BIT                                          E3                          E0914
                                                        70J32                        E0915
                                                        52H0      J32                E0916
E10, LOCATES CELL BEFORE SYMBOL      1                                               E1001
(1) ON LIST (0). SETS H5 + ON        1                                               E1002
LOCATE,-ON NO LOCATE. LOCATES        1                                               E1003
FIRST OCCURRANCE ONLY.               1                                               E1004
                                       E10       J51     91                        E1005
                                       91      11W0                                E1006
                                                 J81                                E1007
                                               70J31                                E1008
                                               11W1                                 E1009
                                                 J2                                 E1010
                                               70        90                         E1011
                                                 J190    91                         E1012
                                       90      11W0      J31                        E1013
E11, MOVES A SYMBOL UP ONE           1                                               E1101
POSITION ON A LIST. INPUT(0) =       1                                               E1102
CELL NAME IMMEDIATELY PRECEDING      1                                               E1103
REQUIRED SYMBOL.                     1                                               E1104
                                       E11     40H0                                 E1105
                                                 J130                               E11051
                                               70        J8                         E11052
                                               40H0                                 E11053
                                               12H0                                 E1106
                                               20K4                                 E1107
                                                 J81                                E1108
                                                 J6                                 E1109
                                                 J50                                E1110
                                               21W0                                 E1111
                                               11K4                                 E1112
                                                 J190                               E1113
                                               21W0      J30                        E1114
E12,                                   E12       J60     J60                        E1201
                                       E13     60*0                                 E1301
                                               52H0                                 E1302
                                                 J60                                E1303
                                               21*0        0000                     E1304
F0,EPAM TOTAL ASSOLIATIONAL          1                                               F0001
LEARNING PROCESS. GIVEN TWO          1                                               F0002
STIMULI (ITEMS),(0) AND(1), AND      1                                               F0003
A NET (2), THIS PROCESS LEARNS       1                                               F0004
(0) AND (1) IN THE NET, AND LINKS    1                                               F0005
FROM (0) TO (1)                      1                                               F0006
                                        F0       J52                               F0007
                                               11W2                                 F0008
                                               11W0                                 F0009
```

```
                                          F1                              F0010
                                        11W2                              F0011
                                        11W1                              F0012
                                          F1                              F0013
                                        11W2                              F0014
                                        11W1                              F0015
                                        11W0                              F0016
                                          F2      J32                     F0017
F1.EPAM ITEM DISCRIMINATION           1                                   F0101
PROCESS. ADDS AN ITEM                 1                                   F0102
(0) TO DISCRM. NET (1)                1                                   F0103
PERFORMING APPROPRIATE TEST           1                                   F0104
ADDITIONS                             1                                   F0105
                                 F1     J44                              F0106
ITEM TO W0                              J20                               F0107
NET NAME TO W1                        60W1                              F0108
                                      11W0                              F0109
ITEM IS SORTED IN NET                   D2                              F0110
SAVE IMAGE LIST IN W2                 60W2                              F0111
TEST FOR NON-NULL I.L.                  D12                             F0112
BRANCH ON NULL I.L.                   7090                              F0113
                                      11W0                              F0114
                                      11W2                              F0115
MATCH ITEM TO IMAGE.EXTRACT DIFF.       D5                              F0116
SAVE LIST OF DIFF. IN W3              60W3                              F0117
TEST FOR NO DIFFERENCES                 J78                             F0118
TERMINATE ON A MATCH                  70J34          DISC. DONE         F0119
ON A NO-MATCH, USE DIFFERENCES        1                                   F0120
TO CREATE DISCRIMINATING              1                                   F0121
TEST STRUCTURE                        1                                   F0122
CONFUSED TERMINAL                     11Y4                              F0123
DIFFERENCES                           11W3                              F0124
CREATE TESTS                            D9                              F0125
SAVE NAME OF TEST STRUCTURE           20W4                              F0126
                                      11Y4                              F0127
                                      11Y5                              F0128
                                      11W4                              F0129
ADD TEST STRUCTURE TO NET               D10                             F0130
                                      11W1                              F0131
                                      11W0                              F0132
                                        J34                             F0133
                                        F1      0000                    F0134
SUBPROGRAM WHICH PUTS AN              1                                   F0135
ITEM IN NET IN PLACE OF               1                                   F0136
NULL IMAGE LIST                       1                                   F0137
NAME OF CURRENT DICTIONARY       90     11Y3                              F0138
ITEM                                    11W0                              F0139
```

```
                                          D17                              F0140
SAVE                                     60W4                              F0141
GET TERMINAL                             11Y4                              F01411
GET DICTIONARY ENTRY                       A1                              F01412
                                           J11     J34                     F01413
F2,EPAM LINKING PROCESS, BUILDS     1                                      F0201
CUE LIST FROM ITEM (0) TO ITEM      1                                      F0202
(1) IN NET(2). THE ITEMS ARE        1                                      F0203
ASSUMED TO HAVE BEEN DISCRIMINATED  1                                      F0204
IN THE NET. NO OUTPUT.              1                                      F0205
                                F2         J43                             F0206
                                           J22                             F0207
                                         11W2                              F0208
                                         11W0                              F0209
SORT ITEM (0) IN NET                       D2                              F0210
                                         20W3      90                      F0211
                                90       11W3                              F0218
                                         11W1                              F0219
ADD LETTER OF 1W1 TOCUELIST AT 1W3         D8                              F0220
                                         11W3                              F0221
FIND CUE LIST AGAIN                        D7                              F0222
                                         11W2                              F0223
                                           J6                              F0224
SORT CUES,**=PROPOSED IMAGE OF 1W1         D2                              F0225
                                         11W1                              F0226
                                           D15                             F0227
                                         7090      J33                     F0228
F10, THE SUBJECT, LEARNING IN       1                                      F10 01
NET U2, SERIAL ANTICIPATION         1                                      F10 02
GET NEW ITEM FROM MEMORY DRUM           F10  11C0           WINDOW         F10 03
SAVE IN IMMEDIATE MEMORY.                    60M0                         F10 04
                                             10U2                         F10 05
                                               J6                         F10 06
DO I RECOGNIZE THIS ITEM                       D23                        F10 07
                                             7090      97                 F10
                                         97  10U2                         F10
PRODUCE, INTERNALLY, A RESPONSE              11M0                         F10 10
TO THE ITEM                                    D21                        F10 11
SAVE IT IN I.M., M2.                         60M2                         F10 12
WRITE THE RESPONSE                             D1                         F10 13
WAIT FOR NEXT ITEM.                            C1                         F10 14
                                             70F10                        F10
LOOK AT NEW ITEM IN DRUM WINDOW              11C0                         F10 15
SAVE IT IN I.M.                              60M1                         F10 16
                                             11M2                         F10 17
CHECK RESPONSE                                 D27                        F10 18
                                             70        93                 F10
```

```
IF WRONG RESPONSE.                          10U2                    F10 20
DO I RECOGNIZE THE ITEM                     11M1                    F10 21
                                              D23                   F10 22
                                            7095                    F10
                                              98                    F10
YES.                                        10U2                    F10 24
                                            11M1                    F10 25
RELEARN THE ASSOCIATION LINK                11M0                    F10 26
                                              F2            0       F10 27
LEARN THE PAIR OF ITEMS              91     10U2                    F10 28
                                            11M1                    F10 29
                                            11M0                    F10 30
                                              F0            0       F10 31
NO RECOGNITION OF THE                90     10K10                   F10 32
STIMULATING ITEM                            40H0                    F10 33
PRODUCE A NULL RESPONSE                       J152                  F10 34
                                            20M2                    F10 35
WAIT FOR NEXT ITEM                            C1                    F10 36
                                            70F10                   F10
                                            10U2                    F10 37
LOOK AT NEW ITEM FROM DRUM                  11C0                    F10 38
SAVE IT                                     60M1                    F10 39
DO I RECOGNIZE THIS ITEM                      D23                   F10 40
                                            70          94          F10
                                              92                    F10
                                            11M0                    F10 42
IS FIRST ITEM AN ANCHOR POINT                 A9                    F10 43
                                            70          910         F10
IS SECOND ITEM AN ANCHOR POINT              11M1                    F10 45
                                              A9                    F10 46
                                            70      0               F10
                                            30H0        91          F10 471
                                     910    30H0        91          F10
                                     92     10K10       J152        F10
                                     94     10U2                    F10
                                            11M1                    F10
                                              D24                   F10
                                            30H0        91          F10
                                     98     10U2                    F10
                                            11M1                    F10
                                              D24       J8          F10
                                     95     0492        91          F10
                                     93     04C6                    F10
                                            11C0                    F10
FORMAT OF AN ITEM, I.E. LETTER,      1                              ITEM001
SYLLABLES, IMAGES,ETC., SHOWING      1                              ITEM002
TYPICAL ITEM.                        1                              ITEM003
```

```
9-0 IS DESC. LIST FOR ITEM SO          1 SO       9   0                    ITEM01
9-1 IS A LINE OF GENERAL               1          9   1                    ITEM02
INFORMATION IN CODED FORM.             1                                   ITEM03
FIRST CODED LETTER OF ITEM             1          9   2                    ITEM04
SECOND CODED LETTER OF ITEM            1          9   3                    ITEM05
THIRD CODED LETTER OF ITEM             1          9   4   0000             ITEM06
EXAMPLE OF A LETTER OF SO              1 9   2      0000                    ITEM07
9-6 IS A LINE OF GENERAL INFO.         1          9   6                    ITEM08
9-7 IS A LIST OF BITS, THUS            1          9   7   0000             ITEM09
                                       1 9   7      0000                    ITEM10
J3 STANDS FOR MINUS                    1          J   3                    ITEM11
                                       1          J   3                    ITEM12
J4 STANDS FOR PLUS                     1          J   4                    ITEM13
                                       1          J   3                    ITEM14
                                       1          J   4                    ITEM15
                                       1          J   4                    ITEM16
                                       1          J   3                    ITEM17
                                       1          J   4   0000             ITEM18
                                         J190   10W0     E13               J19N00
                                         J191   10W1     E13               J19N01
                                         J192   10W2     E13               J19N02
                                         J193   10W3     E13               J19N03
                                         J194   10W4     E13               J19N04
                                         J195   10W5     E13               J19N05
                                         J196   10W6     E13               J19N06
                                         J197   10W7     E13               J19N07
                                         J198   10W8     E13               J19N08
                                         J199   10W9     E13               J19N09
ZERO INTEGER                            +N    0 01 0000   0000             N   01
INTEGER 1                                N    1 01    0   0001             N   02
INTEGER 2                                N    2 01    0   0002             N   03
INTEGER 3                                N    3 01    0   0003             N   04
INTEGER 4                                N    4 01    0   0004             N   05
INTEGER 5                                N    5 01    0   0005             N   06
INTEGER 6                                N    6 01    0   0006             N   07
INTEGER 7                                N    7 01    0   0007             N   08
INTEGER 8                                N    8 01    0   0008             N   09
INTEGER 9                                N    9 01    0   0009             N   10
INTEGER 10                               N   10 01    0   0010             N   11
INTEGER 11                               N   11 01    0   0011             N   12
INTEGER 12                               N   12 01    0   0012             N   13
INTEGER 13                               N   13 01    0   0013             N   14
INTEGER 14                               N   14 01    0   0014             N   15
INTEGER 15                               N   15 01    0   0015             N   16
INTEGER 16                               N   16 01    0   0016             N   17
INTEGER 17                               N   17 01    0   0017             N   18
INTEGER 18                               N   18 01    0   0018             N   19
```

```
INTEGER 19                          N  19 01    0  0019            N   20
INTEGER 20                          N  20 01    0  0020            N   21
NOTICING ORDER FOR LINES, P0        P0       0000                   P0001
                                       N1                          P0002
                                       N3                          P0003
                                       N2      0000                P0004
NOTICING ORDER. P0                  P0        0                    P0001
                                       N1     0                    P0002
NOTICING ORDER. P1                  P1        0                    P0101
                                       N1                          P0102
                                       N2                          P0103
                                       N3                          P0104
                                       N4                          P0105
                                       N5                          P0106
                                       N6                          P0107
                                       N7                          P0108
                                       N8                          P0109
                                       N9                          P0110
                                       N10                         P0111
                                       N11                         P0112
                                       N12                         P0113
                                       N13                         P0114
                                       N14                         P0115
                                       N15     0                   P0116
NO. OF DIFFERENCES EXTRACTED        P2     N1      0     P2=1       P0201
EXAMPLE OF A LETTER CODE         1                                  Q0100
LETTER A                            Q1       0000                  Q0101
                                       90                          Q0102
                                       91      0000                Q0103
                                    90        0000  0000            Q0104
                                    91        0000                  Q0105
                                       J3                          Q0106
                                       J4                          Q0107
                                       J4                          Q0108
                                       J4                          Q0109
                                       J3                          Q0110
                                       J3                          Q0111
                                       J4                          Q0112
                                       J4                          Q0113
                                       J4                          Q0114
                                       J3                          Q0115
                                       J4                          Q0116
                                       J3                          Q0117
                                       J3                          Q0118
                                       J3      0000                Q0119
SORTING NET FOR CAPITAL LETTERS     U   1   9  30                   U01001
LOCALS 9-1 TO 9-110                        Z   0                   U01002
```

```
                9    1          -          U01003
                9    2    0000  +          U01004
     9    1     9   31                     U01005
                Z    0                     U01006
                9    3          -          U01007
                9    4    0000  +          U01008
     9    2     9   32                     U01009
                Z    0                     U01010
                9    5          +          U01011
                9    6    0000  +          U01012
     9    3     9   33                     U01013
                Z    0                     U01014
                9    7          -          U01015
                9    8    0000  +          U01016
     9    4     9   34                     U01017
                Z    0                     U01018
                9    9          -          U01019
                9   10    0000  +          U01020
     9    5     9   35                     U01021
                Z    0                     U01022
                9   11          -          U01023
                9   12    0000  +          U01024
     9    6     9   36                     U01025
                Z    0                     U01026
                9   13          -          U01027
                9   14    0000  +          U01028
     9    7     9   37                     U01029
                Z    0                     U01030
                9   15          -          U01031
                9   16    0000  +          U01032
     9    8     9   38                     U01033
                Z    0                     U01034
                9   17          -          U01035
                9   18    0000  +          U01036
     9    9     9   39                     U01037
                Z    0                     U01038
                9   19          -          U01039
                9   20    0000  +          U01040
     9   10     9   40                     U01041
                Z    0                     U01042
                9   21          -          U01043
                9   22    0000  +          U01044
     9   11     9   41                     U01045
                Z    0          -          U01046
                9   23          +          U01047
                9   24    0000  +          U01048
     9   12     9   42                     U01049
```

```
                    Z    0                    U01050
                    9   25          -         U01051
                    9   26   0000   +         U01052
 9   13             9   43                    U01053
                    Z    0                    U01054
                    9   27          -         U01055
                    9   28   0000   +         U01056
 9   14             9   44                    U01057
                    Z    0                    U01058
                    9   29          -         U01059
                    9   80   0000   +         U01060
 9   15             9   45                    U01061
                    Z1                        U01 062
                    9   81          -         U01063
                    9   82   0000   +         U01064
 9   16             9   46                    U01065
                    Z    0                    U01066
                    9   83          -         U01067
                    9   84   0000   +         U01068
 9   17             9   47   0000             U01069
 9   18             9   48   0000             U01070
 9   19             9   49                    U01071
                    Z    0                    U01072
                    9   85          -         U01073
                    9   86   0000   +         U01074
 9   20             9   50                    U01075
                    Z    0                    U01076
                    9   87          -         U01077
                    9   88   0000   +         U01078
 9   21             9   51   0000             U01079
 9   22             9   52                    U01080
                    Z    0                    U01081
                    9   89          -         U01082
                    9   90   0000   +         U01083
 9   23             9   53                    U01084
                    Z    0                    U01085
                    9   91          -         U01086
                    9   92   0000   +         U01087
 9   24             9   54   0000             U01088
 9   25             9   55   0000             U01089
 9   26             9   56   0000             U01090
 9   27             9   57   0000             U01091
 9   28             9   58   0000             U01092
 9   29             9   59                    U01093
                    Z    0                    U01094
                    9   93          -         U01095
                    9   94   0000   +         U01096
```

```
                        9  80    9  60                          U01097
                                 Z   0                          U01098
                                 9  95          -               U01099
                                 9  96  0000    +               U01100
                        9  81    9  61  0000                    U01101
                        9  82    9  62  0000                    U01102
                        9  83    9  63  0000                    U01103
                        9  84    9  64  0000                    U01104
                        9  85    9  65                          U01105
                                 Z   0                          U01106
                                 9  97          -               U01107
                                 9  98  0000    +               U01108
                        9  86    9  66  0000                    U01109
                        9  87    9  67  0000                    U01110
                        9  88    9  68                          U01111
                                 Z   0                          U01112
                                 9  99          -               U01113
                                 9 100  0000    +               U01114
                        9  89    9  69  0000                    U01115
                        9  90    9  70  0000                    U01116
                        9  91    9  71  0000                    U01117
                        9  92    9  72  0000                    U01118
                        9  93    9  73  0000                    U01119
                        9  94    9  74  0000                    U01120
                        9  95    9  75  0000                    U01121
                        9  96    9  76  0000                    U01122
                        9  97    9  77  0000                    U01123
                        9  98    9  78  0000                    U01124
                        9  99    9  79  0000                    U01125
                        9 100    9 110  0000                    U01126
D.L. FOR UI             9  30     0000                         U01127
                                 A   0                          U01128
                                 J   4                          U01129
                                 A   5                          U01130
                                 N1
                                 A   4                          U01132
                                 N   1  0000                    U01133
D.L. FOR 9-1            9  31     0000                         U01134
                                 A   0                          U01135
                                 J   4                          U01136
                                 A   4                          U01137
                                 N1
                                 A   5                          U01139
                                 N   4  0000                    U01140
D.L. FOR 9-2            9  32     0000                         U01141
                                 A   0                          U01142
                                 J   4                          U01143
```

```
                                A    4                        U01144
                                N1
                                A    5                        U01146
                                N    2   0000                 U01147
U                          9    33   0000                     U01148
                                A    0                        U01149
                                J    4                        U01150
                                A    4                        U01151
                                N1
                                A    5                        U01153
                                N    5   0000                 U01154
                           9    34   0000                     U01155
                                A    0                        U01156
                                J    4                        U01157
                                A    4                        U01158
                                N1
                                A    5                        U01160
                                N    5   0000                 U01161
                           9    35   0000                     U01162
                                A    0                        U01163
                                J    4                        U01164
                                A    4                        U01165
                                N1
                                A    5                        U01167
                                N    3   0000                 U01168
                           9    36   0000                     U01169
                                A    0                        U01170
                                J    4                        U01171
                                A    4                        U01172
                                N1
                                A    5                        U01174
                                N    3   0000                 U01175
                           9    37   0000                     U01176
                                A    0                        U01177
                                J    4                        U01178
                                A    4                        U01179
                                N1
                                A    5                        U01181
                                N    6   0000                 U01182
                           9    38   0000                     U01183
                                A    0                        U01184
                                J    4                        U01185
                                A    4                        U01186
                                N1
                                A    5                        U01188
                                N    8   0000                 U01189
                           9    39   0000                     U01190
```

```
                    A    0                        U01191
                    J    4                        U01192
                    A    4                        U01193
                    N1
                    A    5                        U01195
                    N    7   0000                 U01196
9   40               0000                        U01197
                    A    0                        U01198
                    J    4                        U01199
                    A    4                        U01200
                    N1
                    A    5                        U01202
                    N    7   0000                 U01203
9   41               0000                        U01204
                    A    0                        U01205
                    J    4                        U01206
                    A    4                        U01207
                    N1
                    A    5                        U01209
                    N    5   0000                 U01210
9   42               0000                        U01211
                    A    0                        U01212
                    J    4                        U01213
                    A    4                        U01214
                    N1
                    A    5                        U01216
                    N    4   0000                 U01217
9   43               0000                        U01218
                    A    0                        U01219
                    J    4                        U01220
                    A    4                        U01221
                    N1
                    A    5                        U01223
                    N    4   0000                 U01224
9   44               0000                        U01225
                    A    0                        U01226
                    J    4                        U01227
                    A    4                        U01228
                    N1
                    A    5                        U01230
                    N    5   0000                 U01231
9   45               0000                        U01232
                    A    0                        U01233
                    J    4                        U01234
                    A    4                        U01235
                    N1
                    A    5                        U01237
```

```
            N7          0          U01 238
9   46       0000                  U01239
            A    0                 U01240
            J    4                 U01241
            A    4                 U01242
            N1
            A    5                 U01244
            N   13   0000          U01245
9   47       0000                  U01246
            A    0                 U01247
            J    3                 U01248
            A    1                 U01249
            X    5   0000          U01250
9   48       0000                  U01251
            A    0                 U01252
            J    3                 U01253
            A    1                 U01254
            X   11   0000          U01255
9   49       0000                  U01256
            A    0                 U01257
            J    4                 U01258
            A    4                 U01259
            N1
            A    5                 U01261
            N   13   0000          U01262
9   50       0000                  U01263
            A    0                 U01264
            J    4                 U01265
            A    4                 U01266
            N1
            A    5                 U01268
            N    8   0000          U01269
9   51       0000                  U01270
            A    0                 U01271
            J    3                 U01272
            A    1                 U01273
            X    9   0000          U01274
9   52       0000                  U01275
            A    0                 U01276
            J    4                 U01277
            A    4                 U01278
            N1
            A    5                 U01280
            N    8   0000          U01281
9   53       0000                  U01282
            A    0                 U01283
            J    4                 U01284
```

```
                A    4                                    U01285
                N1
                A    5                                    U01287
                N    6   0000                             U01288
 9   54          0000                                     U01289
                A    0                                    U01290
                J    3                                    U01291
                A    1                                    U01292
                X    3   0000                             U01293
 9   55          0000                                     U01294
                A    0                                    U01295
                J    3                                    U01296
                A    1                                    U01297
                X   17        0                           U01298
 9   56          0000                                     U01299
                A    0                                    U01300
                J    3                                    U01301
                A    1                                    U01302
                X   15   0000                             U01303
 9   57          0000                                     U01304
                A    0                                    U01305
                J    3                                    U01306
                A    1                                    U01307
                X   10   0000                             U01308
 9   58          0000                                     U01309
                A    0                                    U01310
                J    3                                    U01311
                A    1                                    U01312
                X   21   0000                             U01313
 9   59          0000                                     U01314
                A    0                                    U01315
                J    4                                    U01316
                A    4                                    U01317
                N1
                A    5                                    U01319
                N    8   0000                             U01320
 9   60          0000                                     U01321
                A    0                                    U01322
                J    4                                    U01323
                A    4                                    U01324
                N1
                A    5                                    U01326
                N    7   0000                             U01327
 9   61          0000                                     U01328
                A    0                                    U01329
                J    3                                    U01330
                A    1                                    U01331
```

```
            X   6   0000          U01332
9    62      0000                 U01333
            A   0                 U01334
            J   3                 U01335
            A   1                 U01336
            X  12   0000          U01337
9    63      0000                 U01338
            A   0                 U01339
            J   3                 U01340
            A   1                 U01341
            X  26   0000          U01342
9    64      0000                 U01343
            A   0                 U01344
            J   3                 U01345
            A   1                 U01346
            X  14   0000          U01347
9    65      0000                 U01348
            A   0                 U01349
            J   4                 U01350
            A   4                 U01351
            N1
            A   5                 U01353
            N  10   0000          U01354
9    66      0000                 U01355
            A   0               U U01356
            J   3               U U01357
            A   1               U U01358
            X  13   0000        U U01359
9    67      0000               U U01360
            A   0                 U01361
            J   3                 U01362
            A   1                 U01363
            X  20   0000          U01364
9    68      0000                 U01365
            A   0                 U01366
            J   4                 U01367
            A   4                 U01368
            N1
            A   5                 U01370
            N   3   0000          U01371
9    69      0000                 U01372
            A   0                 U01373
            J   3                 U01374
            A   1                 U01375
            X   8   0000          U01376
9    70      0000                 U01377
            A   0                 U01378
```

		J	3		U01379
		A	1		U01380
		X	24	0	U01381
9	71		0000		U01382
		A	0		U01383
		J	3		U01384
		A	1		U01385
		X	7	0000	U01386
9	72		0000		U01387
		A	0		U01388
		J	3		U01389
		A	1		U01390
		X	19	0000	U01391
9	73		0000		U01392
		A	0		U01393
		J	3		U01394
		A	1		U01395
		X	16	0000	U01396
9	74		0000		U01397
		A	0		U01398
		J	3		U01399
		A	1		U01400
		X	18	0000	U01401
9	75		0000		U01402
		A	0		U01403
		J	3		U01404
		A	1		U01405
		X	4	0000	U01406
9	76		0000		U01407
		A	0		U01408
		J	3		U01409
		A	1		U01410
		X	2	0000	U01411
9	77		0000		U01412
		A	0		U01413
		J	3		U01414
		A	1		U01415
		X	23	0000	U01416
9	78		0000		U01417
		A	0		U01418
		J	3		U01419
		A	1		U01420
		X	22	0000	U01421
9	79		0000		U01422
		A	0		U01423
		J	3		U01424
		A	1		U01425

```
                        X  25   0000            U01426
                9 110    0000                   U01427
                         A   0                  U01428
                         J   3                  U01429
                         A   1                  U01430
                         X   1   0000           U01431
                U0002    90000                  U02 01
                         Z0000                  U02 02
                         90001                  U02 03
                         90002   0000           U02 04
                90001    90003   0000           U02 05
                90002    90004   0000           U02 06
                90003     0000                  U02 07
                         A0000                  U02 08
                         J0003                  U02 09
                         A0001                  U02 10
                         X31         0          002
                90004     0000                  U02 12
                         A0000                  U02 13
                         J0003                  U02 14
                         A0001                  U02 15
                         X32         0          002
                90000     0000                  U02 19
                         A0000                  U02 20
                         J0004                  U02 21
                         A0004                  U02 22
                         N0001                  U02 23
                         A0005                  U02 24
                         N0001   0000           U02 25
DICTIONARY OF LETTERS    X   0   9   0   0000   X00001
                9   0     0000                   X00002
                         X   1                  X00003
                         L   1                  X00004
                         X   2                  X00005
                         L   2                  X00006
                         X   3                  X00007
                         L   3                  X00008
                         X   4                  X00009
                         L   4                  X00010
                         X   5                  X00011
                         L   5                  X00012
                         X   6                  X00013
                         L   6                  X00014
                         X   7                  X00015
                         L   7                  X00016
                         X   8                  X00017
                         L   8                  X00018
```

```
                                                X    9                    X00019
                                                L    9                    X00020
                                                X   10                    X00021
                                                L   10                    X00022
                                                X   11                    X00023
                                                L   11                    X00024
                                                X   12                    X00025
                                                L   12                    X00026
DICTIONARY                                      X   13                    X00027
                                                L   13                    X00028
                                                X   14                    X00029
                                                L   14                    X00030
                                                X   15                    X00031
                                                L   15                    X00032
                                                X   16                    X00033
                                                L   16                    X00034
                                                X   17                    X00035
                                                L   17                    X00036
                                                X   18                    X00037
                                                L   18                    X00038
                                                X   19                    X00039
                                                L   19                    X00040
                                                X   20                    X00041
                                                L   20                    X00042
                                                X   21                    X00043
                                                L   21                    X00044
                                                X   22                    X00045
                                                L   22                    X00046
                                                X   23                    X00047
                                                L   23                    X00048
                                                X   24                    X00049
                                                L   24                    X00050
                                                X   25                    X00051
DICTIONARY                                      L   25                    X00052
                                                X   26                    X00053
                                                L   26   0000             X00054
                                X1              X0                  0
                                X2              X0                  0
                                X3              X0                  0
                                X4              X0                  0
                                X5              X0                  0
                                X6              X0                  0
                                X7              X0                  0
                                X8              X0                  0
                                X9              X0                  0
                                X10             X0                  0
                                X11             X0                  0
```

```
                                X12      X0          0
                                X13      X0          0
                                X14      X0          0
                                X15      X0          0
                                X16      X0          0
                                X17      X0          0
                                X18      X0          0
                                X19      X0          0
                                X20      X0          0
                                X21      X0          0
                                X22      X0          0
                                X23      X0          0
                                X24      X0          0
                                X25      X0          0
                                X26      X0          0
                                X30      90          0
                                90               0
                                         X31.
                                         K10
                                         X32
                                         K10         0
                                X31      X30         0
                                X32      X30         0
                                Y3       X30         0
Z0, TEST I SUB J = 0, SET H5+   Z   0 40W   4                    Z00 01
ON FINDING J4,- ON FINDING              40W   5                    Z00 02
J3. NAME OF STIMULUS BEING              11W   2                    Z00 03
SORTED ALWAYS IS 1Y1, NODE              10A   4                    Z00 04
OF THE TEST IS 1W2. WHEN                  J  10                    Z00 05
I SUB J CANNOT BE FOUND, H5 IS          70J   7          ES        Z00 06
SET RANDOMLY.                           20W   4                    Z00 07
                                        11W   2                    Z00 08
                                        10A   5                    Z00 09
                                          J  10                    Z00 10
                                        70J   7          ES        Z00 11
END OF SEARCH FOR INPUTS                20W   5                    Z00 12
                                        11Y   1                    Z00 13
                                        11W   4                    Z00 14
LOCATES LINE OF CODE                      E   3                    Z00 15
                                        709   0                    Z00 16
                                        52H   0                    Z00 161
                                          J82                      Z00162
                                        11W   5                    Z00 17
LOCATES BIT OF CODE                       E   3                    Z00 18
                                        709   0                    Z00 19
                                        52H     91                 Z00 20
                                9   1   J   1                      Z00 21
```

```
      30W   4                    Z00 22
      30W   5  0000              Z00 23
9  0   E   5 9   1               Z00 24
Z1      Z0    J5                 Z01 01
```