SYSTEM RESEARCH LTD

FINAL SCIENTIFIC REPORT

Contract F 61 052 67 C 0010 1 Sep. 1966 - 30 Nov. 1968

Gordon Pask

June, 1969

20 HILL RISE
RICHMOND
SURREY
ENGLAND
RIC 0801

1. This document has been approved for public release and sale; its distribution is unlimited.

Contract No:

F61 052 67 C 0010

June, 1969

FINAL SCIENTIFIC REPORT

"Cybernetic Models for Learning"

LEARNING STRATEGIES AND TEACHING STRATEGIES

Gordon Pask,

System Research Ltd.,
20, Hill Rise,
Richmond, Surrey,
England.

"THE RESEARCH REPORTED IN THIS DOCUMENT HAS BEEN SPONSORED IN WHOLE, OR IN PART, BY THE AIR FORCE OFFICE OF SCIENTIFIC RESEARCH, UNDER CONTRACT NO. F 61 052 67 C 0010, THROUGH THE EUROPEAN OFFICE OF AEROSPACE RESEARCH (OAR) UNITED STATES AIRFORCE."

Final Scientific Report of Contract F61-052-67-C-0010
"Cybernetic Models for Learning"

LEARNING STRATEGIES AND TEACHING STRATEGIES

Gordon Pask, G.L. Mallen, B. Scott, D. Watts, C. Whitehouse
System Research Ltd.

ABSTRACT:

Human learning strategies are considered partly for their own sake but chiefly with a view to designing good teaching strategies to build into an adaptive (CAI) teaching system.

In pursuit of the investigation, we introduced a new experimental method (EXPCI) for externalising the learning strategy adopted by a free learning subject, and we developed a new empirical measure of uncertainty. A new procedure for response analysis is also described.

Hypotheses about the nature of the learning process in a rule application task are crystallised in a computer simulated learning model, which is thoroughly documented. The "conversational" adaptive teaching system, used in the main experiments, is designed on the basis of this model.

Experiments have been carried out using human subjects in a rule application task to compare free learning with learning controlled by this and other teaching systems. Quite unequivocally, learning is more effective when it is controlled by the adaptive system. Further, a system which employs a "compromise" teaching strategy (arrived at by discourse with the student) is more effective than a rigid, single strategy, system. In an attempt to explain why t is is so, a group of hypotheses were tested and enough of them have been emprically verified to provide a clear picture of how learning takes place (and fails to take place) in this particular experimental situation.

"LEARNING STRATEGIES AND TEACHING STRATEGIES"

List of Contents

SECTION : 1

INTRODUCTION AND AN ACCOUNT OF THE EXPERIMENTAL METHODS

1.1. Objectives

This study of learning strategies and learning set has two parts, namely (1) A part involving experiments with real life subjects and (2) a part concerned with a computer simulated model of the learning, which, when interpreted, constitutes a theory of learning.

In the experimental part, we have emphasised two aspects of the field (I) How a subject directs his attention and explores the problem environment created by the task to be learned and (II) How the subjects learning process can be facilitated by conversational, or partially cooperative, teaching systems (in practice, CAI systems). All of the experiments have been carried out in such a way that the subject interacts with an automaton responsible for controlling the experimental environment. In some experimental conditions, the automaton is a procedural device; it imposes behavioural constraints that represent the rules of the task, but it allows the subject a great deal of freedom, especially in his choice of a strategy. In other conditions, the automaton is either a simple adaptive teaching machine or a conversational machine.

In order to teach (i.e. to control a learning process), it is necessary to have a model for this process. The computer simulation is a model of this sort. It contains one subprogramme representing the subject and another subprogramme representing the automaton and these subprogrammes interact.

The gross action of the metasystem can be explained within the theory of self-organising systems, considered in Section 3 and Section 4. However, an adequate account of the mechanisms involved (or an adequate design paradigm) calls for a more detailed model, which is provided by the computer programme of Section 5.

In the course of the experiments, it became evident (1) that different subjects use different strategies for learning the same skill and (2) that these strategies may or may not be fitted to aspects of mental organisation of which the subject is often unperfectly aware. It, thus, seemed likely that a competent teaching system should (a) be based upon a class of possible teaching strategies rather than a single strategy, (b) have a facility for selecting one strategy or another as a result of higher level (meta) discourse with the student, and, (c) contain a monitoring facility for ensuring that the chosen strategy is appropriate to the individual's mental organisation. Such a teaching system is an adaptive metasystem in the sense of Ref.(1).

The experimental data supports the cogency of points (a), (b) and (c); it is also possible to assert, with considerable confidence, that an adaptive metasystem is a better instructional instrument (for this skill) than either the attention directing process in a free learning subject or a more restricted, single strategy, teaching system.

1.2. Historical Review and Background Material

In our previous work on teaching and learning we have used "rule application" as the skill to be instructed (mainly by adaptively controlled methods). Details of the techniques employed and the results obtained appear in a series of publications (the key papers being references (1) and (2)). The general form of the task is indicated in Fig.1. For various skills coming under the rubric "rule application", adaptive training methods were shown to have a clear advantage over plausible and intuitively effective non adaptive methods; this, perhaps, was the primary result. In addition, however, we accumulated a good deal of "know how" about the behavioural and cognitive processes that go on whilst the subject is learning. Thus, when it came to a study of learning and teaching strategies we were anxious to use a similar task environment; (1) in order to obtain grossly comparable data (a relatively minor point) and (2) because of our quite detailed insight into the learning process itself.

The rule application task has much to recommend it as a framework within which to experiment. In the simplest situation, there is a single rule which relates singly occurring stimulus lamps to single response keys (Fig.1); the resulting task is thus equivalent to paired associate learning, since the lamps and the keys could easily be labelled with nonsense syllables. Using more than one rule (and alternating the rules in a prescribed fashion) the task resembles multiple list learning. The different rules can be chosen, at the experimenter's discretion, to be entirely dissimilar, to have deceptively similar parts (studies of interference) or to have genuinely similar parts (studies of positive transfer). If the rule can be easily described, it is possible to investigate either concept learning (the rule is unstated and must be learned by experience) or concept utilization (a description of the rule is

presented to the subject at the outset). Insofar as there are describable relationships between the different rules employed in an experiment, it is also possible to study higher level concept attainment (or utilisation). Of course, the experimenter is not restricted to rules entailing a one to one correspondence between the lamps and the response keys. Though rules of this type have been used in all of the main experiments, pilot investigations with many to one mappings yielded quite interesting results. In all this, stimuli are quite easy to deliver and responses and their latencies are readily detectable.

The basic task may be elaborated,(and for most of the experiments has been elaborated) by presenting stimuli, which consist in a group of simultaneously illuminated stimulus lamps. In this case, the correct response involves a group of key pressing operations; with a one to one rule as many key depressions as there are lamps in the stimulus. Since each operation has a separately determinable latency, the experimenter is in a position to observe response patterns consisting of several response components, which may either be produced together or in a sequence. Hence, quite a lot of the subject's data processing is externalised in the form of the response. The possibility of obtaining this much information about the "Black Box" is a crucial feature of the experiments to be described, or any like them.

It is usual to employ 4 lamp problems (Ref. (2)) or, sometimes, in rule alternation, 3 lamp problems (Ref. (1)). If 4 lamp problems are regarded as the "real" problems (the subject is given the goal of solving a sequence of 4 lamp problems) then 3 lamp, 2 lamp and 1 lamp problems are <u>subproblems</u> and the goal of dealing with a sequence of 3, 2 or 1 lamp problems is a <u>subgoal</u> of the original goal.

The task environment sketched in Fig.1 has a deceptively simple appearance but is, in fact, quite rich in possibilities. A great deal of this richness is due to the time constraint cited in Fig.1 (i.e. that the subject is forced to respond within a time interval of Δt after the presentation of a stimulus, if he is to achieve a correct response). This constraint is not artificial. All problems are posed with reference to a time allowed for their solution. Often, of course, this is a long time. But, in the present case, the interval Δt is chosen so as to prohibit the obvious expedient of recalling the S-R-connections given by the rule and executing the responses operations one after another. In order to succeed, the subject must conceive the skill in some way and build up perceptual motor groupings appropriate to this framework.

Hence, although the task looks superficially like a stimulus,response, association task°, it really has a complex structure and there is ample evidence that the majority of subjects view it as an exercise in problem solving (even in the early experiments, where, if anything, the subjects were given a "conditioning" or "S-R" orientation by the nature of the instructions). In

---

FOOTNOTE: °Whatever this means. The phrase is quite often used but it seems to reflect little more than a professional disposition to look at statistically regular and (from the present viewpoint, rather uninteresting) facets of mental activity. Consider ordinary paired associate learning, for example. Of course, the mean learning curve etc. can be predicted on the basis of a statistical model in which unitary stimuli become "associated" with unitary responses. But anyone who has learned a "paired associate" list is aware of the cognitive processing which invariably goes on and is, of course, obscured in a purely behaviouristic analysis of the process. In contrast, we are interested (1) in this cognitive processing and (2) in the information processing models such as Atkinson and Shifframs model or Fiegenbaum's EPAM, that bridge the gap between cognitive and S-R theory.

general, the stimuli are interpreted as designating problems and responses are conceived as the result of problem solving operations. This is partly a comment about the subject's attitudes and partly a reflection of the fact that it would be difficult to interpret the complex response latency data without invoking a problem solving process. Of course, the problem solving may be unconscious. It is probably fair to say that the subject feels he is in a problem solving situation but is imperfectly aware of the process whereby he solves individual problems. In general, also, the subject adopts cognitive representations of the problem and uses plans or strategies as part of his learning set (once again a body of introspective data is supported by an inspection of attention directing behaviour, pauses and intermittencies).

From the experimenter's point of view it is quite possible to examine the structure of the problem solving (response producing) process on a convenient time scale. The various components of the (generally complex) response are produced in regular patterns; for example, in one mode of activity pairs of response components are produced almost simultaneously, i.e. the occurrence times for each member of the pair are no more than 20m. secs. apart. After studying the simultaniety, or otherwise, of the components and the order in which components or groups of them occur, it is possible to make a number of inferences about the generating process. Further, fairly complex properties of this process can be detected and used as control variables for a teaching strategy.

1.3. Development of the Experimental Environment

As above, the original experiments suggested that a great deal of cognitive and strategic activity goes on as the rule application skill is learned. However, apart from introspection and gross behavioural indices, they incorporated no easy method for externalising this activity. The present experiments were performed in an isomorphic task environment but did; amongst other things, provide means for externalising the mental processes.

In order to obtain these facilities, it was first necessary to impose a definite cognitive structure upon the task and to provide a language for describing it and making prescriptive statements or plans; (for example, statements about which part of, or subproblem in, the environment would be attended to at a given instant; plans for exploration of the environment). These matters are discussed in Section 1.4. and Section 1.5.

Next, given the structure and the language, it was necessary to set up conditions in which the subject would use the language and thus externalise his mental processes in discourse with the experimenter (or a control mechanism). This question is considered in Section 1.6.

Amongst other things, the subject can externalise his mental state by making assertions (in the given language) about what he definitely intends to do (by way of subproblem class exploration) and probabilistic statements about his degree of belief in selections or courses of action. The relevant probabilistic response techniques are considered in Section 1.7. and the extraction of behavioural indices in Section 1.8.

1.4. Imposing a Cognitive Structure on the Task

Consider the set of 4 lamp stimuli, which can be selected from amongst 8 lamps: a, b, c, d, e, f, g, h. In the previous experiments, the stimulus sequence consisted in 8 items with the property that each lamp occurred with equal frequency and that even frequency distributions were maintained over subsets of 3, 2 and 1 lamp stimuli posing simplified problems. The arrangement concerned is shown in Fig.2.

For the present purpose we wished to impose a cognitive structure upon the skill. One way of doing so (and the way actually chosen) is shown in Fig.3. The stimulus lamps now designate the values of binary attributes A, B, C and D; thus,

lamp a denotes the value "1" of A and lamp e denotes the value of "0" of A; since A must have one value or the other, and since it cannot have both values simultaneously, a constraint is introduced, i.e. that a and e may not be simultaneously illuminated, for one designates the complement of the other. Similar comments apply to the attribute B and lamps b and f, to attribute C and lamps c and g, and to attribute D and lamps d and h. There are clearly 16 different 4 lamp stimuli which satisfy this constraint, the stimuli appearing in Fig.2 are 8 of them. The entire set of 4 lamp stimuli admissible when the attribute value constraint is applied, is exhibited in Fig.4 in the form of codes that generate the stimuli by turning on lamps a, b, c, d, e, f, g, h, (where "+" is "on" and "-" is "off"). The code set is called the stimulus generating set and is divisible into 8 membered subsets such as α and β. Fig.4 also shows the values of "1" or "0" assumed by the attributes A,B,C,D, when each of these 4 lamp stimuli is presented by selecting a stimulus code and using it as an input to the display mechanism.

In order to bring these constraints home to the subject, the display and response board is modified. The new configuration is shown in Fig.5. Each complementary pair of stimulus lamps is clearly demarcated and associated with one of the attributes A,B,C,D, (the differently coloured lamps in each pair of Fig.5 refer to two different rules which may be in use. However, unless there is a statement to the contrary, we shall assume that only one rule is in use and, thus, that only one colour of lamp is employed).

The conjoint values of the attributes A,B,C,D, determine points in a signal space; these points are designated by the stimuli generated from the generating set and the stimuli pose problems. Now it is possible to determine an hierarchy of sub-problems by regarding one or more of the attributes as null or

irrelevant (say of value ϕ). Formally, the expedient of asserting that some of the attributes may be irrelevant determines a set of subspaces of the signal space. More cogently, it gives rise to the subsets of stimuli exhibited in the partially ordered structure of Fig.6 as X_{ABCD}, X_{ABC}...., X_{AB}...., X_{A}...., in the sense that selecting all stimulus codes in the generating set will (according to the attributes deemed to be irrelevant) give rise to stimuli X of Fig.7, with the property that $x \in X_i$ poses a subproblem of type i (unless it happens that i = ABCD, when the subproblem posed is a full problem). Similarly, if α or β is chosen, rather than the entire stimulus generating set, then a subset of derived stimuli is produced (all of them generating subproblems of the appropriate type). Moreover, in the case of α and β all of the subproblems are posed with equal relative frequencies (this is not true of all subsets of the stimulus generating set). α and β happen to be <u>symmetrical</u>. Certain <u>unsymmetrical</u> stimulus generating subsets have also been employed but attention has been restricted to generating subsets such that all relevant subproblems are posed, perhaps with different relative frequencies, by the stimuli in each of the stimulus subsets.

In the course of an experiment, stimulus codes are selected from the generating subset (say from α) by a sequence of code numbers, such as 8, 1, 2, 5, 4, 7, 3. In general, the stimulus sequence is divided into blocks in each of which each stimulus code is selected at least once. Now the act of selecting a stimulus code gives rise to a derived stimulus, i.e. to <u>the</u> stimulus derived from this 4 lamp stimulus code. However, the number of elements in this stimulus, depends upon the attributes considered relevant or, more succinctly, upon which of the subsets X_i in Fig.6 is selected. The particular derived stimulus can, of course, be read off directly from Fig.7.

To keep the whole "problem to subproblem" relation in the student's mind, the following expedient is adopted; the derived stimulus is displayed against the background of the 4 lamp stimulus from which it is derived. Thus, in Fig.5 if stimulus 7 is selected at the n^{th} trial, and if the subset X_{AB} is selected as relevant, then (I) the subject sees the attribute values A = 0, B = 0, C = 0, D = 1, (or the stimulus b i f g) as a dimly illuminated flashing configuration (II) he sees the relevant attribute values A = 0, B = 0 (or the derived stimulus b d) as a brightly illuminated pair superimposed upon the flashing background.

Stimuli may, thus, be of many sorts, belonging to different subsets X_i. But in all cases they are presented in the context of the stimuli from which they are derived. Of course, if the selected subset is X_{ABCD}, then no flashing lamps will be visible.

To complete the story, selection of a stimulus subset also determines which responses are interpreted as relevant in any assessment of rectitude. It is clear that any one to one rule imposes a partitioning upon the response keys (so that pairs of keys refer to the values of a single attribute). Only the response components in the appropriate n-tuple of keys are referred to in any marking or scoring procedure. For example, a derived stimulus in X_{AB} calls for a response with a pair of components, not 4 of them. Hence, it is realistic to contend that the members of stimulus subsets other than X_{ABCD} do denote subproblems of the main problems.

The physical response arrangements of Fig.5 are the same as those in Fig.1. The key pairs corresponding to the attribute values might, quite reasonably, have been replaced by "attribute response buttons" of the sort used in a much earlier series of experiments on concept acquisition (Ref. 3); one button for each value of the attribute; no button depression signifying irrelevance. This possibility was debated but it was eventually decided to leave the response keys as they stood because with the present arrangement

the movement time is short and substantially the same for each component of the response. Hence, in the latency analysis (which is chiefly intended to classify the ordering and grouping of the response components) it is legitimate to neglect the influence of movement time at the first order of approximation.

1.5. Language Systems and Experimental Discourse

In the course of an experiment the experimenter presents a sequence of stimuli which are interpreted by the subject as problems. Similarly, the subject produces a sequence of responses which the experimenter interprets as problem solving operations. The conduct of this interaction is clearly a discourse and it takes place in a language. This language will be called $\mathcal{L}^0$, the lowest level in a stratified object language $\mathcal{L} = \mathcal{L}^0, \mathcal{L}^1, \ldots$ (in contrast to the metalanguage, $\mathcal{L}^*$, which the experimenter uses to describe the experimental setup, to give instructions, and so on).

Frequently, the object language is glossed as a stimulus response modality and this usage is harmless enough provided that $\mathcal{L} = \mathcal{L}^0$ and the subject is neither encouraged to comment upon nor to modify the environment in the course of the experiment.

In the present case, however, the experimenter is anxious to examine how the subject wants to explore his environment (the set of subproblem classes or stimulus subsets of Fig.6) and in certain circumstances he may allow the subject to engage in active exploration. Because of this a primitive higher level, $\mathcal{L}^1$ of $\mathcal{L}$ must be adjoined to the lowest level object language $\mathcal{L}^0$.

The $\mathcal{L}^0$ level of $\mathcal{L}$ remains quite trivial but should be stated. The subject is required (by the rules of $\mathcal{L}^0$) to consider stimuli as designating problems. His legal expressions in $\mathcal{L}^0$ consist in key pressing responses containing as many components as there are lamps in the stimulus (clearly a correct response is a legal response that also satisfies the prevailing rule).

The l^1 level of l is not quite so obvious. To provide a framework for the l^1 interaction, the stimulus or trial sequence is divided into distinct blocks of trials within any one of which the subject receives a representative sequence of problem posing stimuli (each of the 4 lamp stimulus codes is selected at least once and a possibly simplified stimulus derived from it is displayed). Further, at the end of a trial block, the subject is required to make an l^1 statement of which one of the subsets (in Fig.6) will occupy his attention throughout the subsequent block; that is, a statement of which subgoal he will aim for. In the simplest case, this subgoal statement must be unequivocal and unique, i.e. he must state "I wish to work on block i" (where i indexes one of the subsets in Fig.6). To make this l^1 "statement" he presses any n tuple of the 4 attribute buttons A,B,C,D, shown in Fig.5. (In addition, when more than one rule is employed, he is required to "say" that he will attend to one rule or the other or both of them. This he does by manipulating the rule buttons in Fig.5). Hence, in the simplest case, the subject's utterances consist in configurations of the "attribute buttons" and the "rule buttons".

Now it often happens that the subject entertains considerable doubt about what he should do next. He is literally in the position of saying "I could bet so much on each of several possibilities but I could not select one of them with certainty". To comprehend this situation, l^1 is extended to permit probabilistic statements which, depending upon the experimental conditions employed, may either be mechanised or not. The principle is the same in both cases.

In the unmechanised case, the experimenter asks the subject to state the possibilities he has in mind. If there is only one, then his statement is unequivocal, as before. Generally, when there are several (say m) possibilities, the experimenter uses a simple betting

situation to make the subject state probability estimating numbers r_i such that $\Sigma_i r_i = 1$, $i = 1,....m$. The vector $\vec{r}_i$ is now regarded as a legal $\mathcal{L}^1$ statement.

In the mechanised case, the subject is presented with the $\mathcal{L}^1$ response board of Fig.8 (in place of the attribute buttons of Fig.5). If he opts for a certainty, then he presses one of the 15 buttons on the board shown in Fig.8, together with the certainty button. If he is in doubt, he selects a subset of buttons to designate the set of subproblem classes under consideration. For each button pressed, an indicator lamp is illuminated. For each indicator lamp illuminated, the subject must adjust the betting potentiometers (values of r_i) so that the meter of Fig.8 shows a zero reading (in which case $\Sigma_i r_i = 1$)°. At this point, the subject submits his $\mathcal{L}^1$ statement of r_i by pressing the submit button shown in Fig.8. The statement is interpreted by the experimental control mechanism and the indicator lamps are turned off.

These statements of deterministic or probabilistic intention to attend are the only $\mathcal{L}^1$ statements that the subject is allowed to make in the course of an experiment. Other $\mathcal{L}^1$ statements are made at the beginning of some experiments and when the experiment is completed but these will be considered as "Interrogation Procedures" in the detailed account of the experiments.

FOOTNOTE:

° A similar technique has been employed by Baker[4] to determine subjective degrees of belief with respect to $\mathcal{L}^0$ response alternatives. Baker uses a histogram displayed on the cathode ray tube display of a computer, the lines representing values of r_i. The subject lengthens or shortens the lines using a light pen facility. As he does so the programme automatically normalises the histogram.

The experimenter can also make $\mathcal{L}^1$ statements. Like the subject, his utterances occur at the end of a trial block.

(1) He can say that he is going to present a certain class of subproblems or subproblems throughout the subsequent block and that he will interpret all responses to stimuli as attempted solutions to these problems. According to the conventions of the system, if he says so, then he must do so, i.e. he must select a stimulus subset X_i throughout the subsequent trial block. It will be recalled that the selection of X_i also determines how the subject's responses are interpreted.

(2) The experimenter can deliver $\mathcal{L}^1$ evaluative statements about the subject's performance. The first of these is a block score, displayed via the "score meter" of Fig.5. Several scoring functions have been used, the simplest being the number of correct responses in a block.

(3) Insofar as the subject has generally attended to a whole sequence of subsets, X_i, the experimenter is able to make an $\mathcal{L}^1$ summary statement giving him information about the last value of score achieved with respect to each subset visited in the past. This material is either displayed on cards handed to the subject (Fig.9) or through the "past score" display of Fig.10.

(4) At each trial[6], the experimenter provides knowledge of results information. This (depending upon the experimental conditions employed) may be partial (the entire response is correct or not; displayed via the "All Correct" lamp of Fig.1 and Fig.3) or complete (a particular response component is correct or mistaken; displayed through the "correct component" lamps of Fig.1 or Fig.3).

(5) At each block the experimenter can make a limited number of instructional $\mathcal{L}^1$ statements, which either request the subject to adopt a particular mental set or tell him that he is going to be tested with respect to some property of his behaviour. These instructional statements will be discussed in the context of the teaching systems, where they are actually used.

FOOTNOTE: [6] I must apologise for a slightly barbaric, though consistent, usage of "levels of discourse". Apart from the knowledge of results statements, all $\mathcal{L}^1$ statements are made at the end of a block, whereas these occur at the end of each trial. Further, although score and last score are clearly "evaluations" and thus $\mathcal{L}^1$ statements, "knowledge of results" is usually conceived of as part of the stimulus response modality (thus an $\mathcal{L}^0$ statement). However, it clearly is evaluative and, from the point of view of a control model, it is altogether more convenient and consistent to regard a knowledge of results statement as an $\mathcal{L}^1$ statement since it is a signal at a higher level of control in a control hierarchy.

1.6. Externalising Mentation

The subject may use the object language $L = L^0, L^1$, to interact with whoever or whatever controls the experimental environment, but it does not follow that he will do so; the provision of an object language is no more than a prerequisite for establishing discourse. In order to be sure of a flow of discourse, it is necessary to establish an experimental contract with the subject and for the experimenter to keep his part of the bargain.

By its nature, any contract whatever entails the provision of certain cooperative, or partially cooperative, facilities from which the subject can benefit provided he is aiming for an agreed goal. Hence, the methods employed in maintaining an experimental contract with a view to externalising mentation, are reasonably called "Externalisation by Partially Cooperative Interaction" or "EBPCI" methods.

The exact form of EBPCI depends upon the experimenter's objective. So far as the present study is concerned, two objectives must be considered (I) obtaining information about the subject's learning strategies and (II) teaching him to perform the problem solving skill. In fact, the differences between EBPCI techniques used for (I) and (II) are not very great but this section deals specifically with objective (I) (objective (II) is discussed in the context of teaching systems).

1.6.1. The Automaton

To instrument the EBPCI method, the experiment is controlled by the automaton shown in Fig.11 and flow charted in Fig.12. The functions of this automaton can, in certain instances, be carried out by the experimenter. But if they are, then the experimenter must obey definite rules equivalent to those programmed into the automaton. Consequently, it is convenient to describe the EBPCI method as though the experimenter had delegated his role entirely to the automaton and to make qualifying comments whenever certain operations have, in fact, been performed manually.

The system in Fig.11 and Fig.12 incorporates the cognitive framework of Section 1.4. and the object language $\mathcal{L} = \mathcal{L}^0, \mathcal{L}^1$, of 1.5. These are embodied in several processing mechanisms. There is a facility for selecting stimulus codes from α, β, or any other generating set; a facility for selecting any index $i = A, AB, \ldots$ of a stimulus subset; a mechanism for using the selected stimulus code to produce a stimulus $x(n) \in X_{i(M)}$ at the n^{th} trial in the M^{th} block if $i_{(M)}$ is selected at the beginning of the M^{th} block.

As pointed out in Section 1.4., any stimulus may be represented in terms of its generating code. Thus x(n) is represented by the vector:

$$x(n) = x_1(n) \ldots\ldots\ldots\ldots\ldots\ldots x_8(n)$$

in which, for $j = 1, \ldots\ldots 8$, $x_j(n)$ can assume the values +, −, and $\emptyset$ or "null" (for $x(n) \in X_{ABCD}$ no value of $\emptyset$ appears in the vector but for all stimuli designating subproblems at least a pair of variables assume the null value). Similarly, a response y(n) can be represented in terms of variables denoting its components, namely as:

$$y(n) = y_1(n) \ldots\ldots\ldots\ldots\ldots\ldots y_8(n)$$

where $y_j(n)$ assumes the possible values "+" (pressed key) or "−" (not pressed key). If the rule (in the rule application skill) is Ω, a mapping from $x_1 \ldots\ldots x_8$ onto $y_1 \ldots\ldots y_8$, then the automaton produces an internal representation $\Omega\, x(n)$ of the correct response corresponding to a solution and it compares this with y(n) to obtain a knowledge of results vector $\sigma(n)$ defined as:

$$\sigma(n) = \sigma_0(n);\ \sigma_1(n) \ldots\ldots \sigma_8(n)$$

$$\text{where } \sigma_0(n) = \begin{cases} 1 \text{ if } \bigwedge_i \sigma_i(n) = 1 \\ 0 \text{ if not} \end{cases}$$

$$\text{and where } \sigma_j(n) = \begin{cases} 1 \text{ if } (x_j(n) = y_j(n)) \vee x_j(n) = \emptyset \\ 0 \text{ if not} \end{cases}$$

These operations and the entire conduct of the experiment are sequenced by the automaton, as shown in the operating cycle chart of Fig.13. Since "Blocks" of trials may be defined in at least two different ways (see below) the number of trials in a block is deliberately unstated in Fig. 13.

At the end of each block, a score $\rho(n)$ is computed and the values of the last score function are updated for presentation on cards or via the last score display[0]. At the end of the interval δt (the interval allowed for submitting a response), the automaton delivers a knowledge of results signal. In the partial knowledge of results condition, this is the value of $\delta_0(n)$ (orange lamp if $\delta_0(n) = 1$, blue lamp if $\delta_0(n) = 0$, as in Fig.5). In the complete knowledge of results condition, the automaton displays the values of $\Omega[x(n)]$ together with the values of $y(n)$ via the lamps of Fig.5. Since the j^{th} signal lamp is illuminated only if $x_j(n) = +$ (not if $x_j(n) = \beta$ or $-$), the subject is able to compare what he should have done about the relevant problem or subproblem with what he actually did do. Redundantly, he is also provided with an indication of $\delta_1(n) \ldots \delta_8(n)$.

The remaining functions of the automaton are more conveniently discussed in the context of particular aspects of the EBPCI method.

FOOTNOTE: [0] A notation for the last score function is necessarily complicated and adds nothing to a simple description. Last score is computed by setting 15 triples of lock relays, which energise signal lamps. The i-th triple is set or reset at the end of a block in which X_i has been selected.
The value set (the member of the triple energised) depends upon the value of $\rho(n)$ at this instant.

1.6.2. Preliminary Experience

Let us assume that the subject is familiar with the general type of problem, conceivably by previous experience, in solving rule application problems under a different and less difficult rule. The first step in establishing an experimental contract is to obtain the subject's agreement (1) that (given a fresh rule) he will aim for the goal of learning to solve all 4 lamp problems when these are posed in random order (it will be recalled that an essential part of the problem specification is that a correct response must be produced each Δt secs.). (2) that he will do so in the framework afforded by $\mathcal{L} = \mathcal{L}^0, \mathcal{L}^1$, and its interpretation.

1.6.3. Knowledge of Results

On considering his previous experience, the subject appreciates that he cannot learn to solve the problems at all unless he is provided with knowledge of results statements. The first part of the contract is established by saying "if you will obey the rules of $\mathcal{L}$ and will bother to respond, and thus designate your solution to each problem, the automaton will give you knowledge of results. Unless you do respond, this information cannot be delivered". (Conversely, it would be a waste of effort to respond unless the information was made available). It should be recalled that knowledge of results may either be partial or complete in the present experiment. The form of knowledge of results has an important bearing upon the next part of the contractual agreement.

1.6.4. Attention Directing

The subject is introduced to the part of $\mathcal{L}^1$ in which it is possible to talk about subgoals by selecting stimulus subsets X_i, which designate subclasses of problems. Further, he is told that the automaton will interpret his $\mathcal{L}^1$ statement "I want to deal

with the i^{th} subclass of problems" as an instruction to select X_i and to present some $x(n) \in X_{i(M)}$ throughout the M^{th} block of trials. Finally, the subject is impelled to make one (and only one) l^1 statement evaluating i(M) at the beginning of each block (the automaton waits for him to so so; if he fails to do so after 20 secs. it sounds a buzzer; in fact, our subjects did not dawdle).

Broadly, the subject is encouraged to output a statement of which subproblem class he actually <u>is</u> attending to because doing so allows him to get out of an otherwise overloading situation and to control his uncertainty about how to solve problems. More specifically, the following contract is established. <u>If</u> the subject <u>does</u> state the problem class (by evaluating i(M), then the automaton cooperates in the following ways:

(1) by presenting stimuli denoting subproblems in the context of the 4 lamp problems from which they are derived.

(2) by interpreting the subject's response to each derived stimulus as a response to a subproblem and by marking it as such (in the provision of knowledge of results).

(3) by computing a score $\rho(n)$ over a block of trials addressed to subproblems in the i^{th} class.

(4) by retaining this as a last score associated with the rehearsal of $x \in X_i$.

It should be emphasised that the automaton is unable to cooperate unless the subject does make an $\mathcal{L}^1$ statement evaluating i(M). Conversely, all of (1), (2), (3) and (4) above are of genuine assistance to the subject. His brain, like any other brain, is an imperfect information storage device. The subject needs to keep track of performance information in order to achieve the agreed goal. The automaton cooperates by augmenting his information storage facilities.

1.6.5. Probabilistic Statements

Having agreed to the last clause of the contract, the subject is placed in the position of determining and accepting a sequence of subgoals and rehearsing the solution of subclasses of the main problems. Further, one subgoal or subclass must be selected for each block of trials. In reality, however, the subject often has uncertainty about what to do; i.e. he entertains varying degrees of belief in the cogency of different subgoals rather than an outright conviction that one subgoal is appropriate.

We should like to externalise these degrees of belief in terms of an $\mathcal{L}^1$ probability estimating vector, $\vec{r}_i$. But why should the subject be bothered to construct this vector as an $\mathcal{L}^1$ statement. Since one and only one subgoal must be selected for each block of trials, it wouldbe far easier just to state the most probable choice as a certainty and have done with it.[6]

FOOTNOTE:

[6] If that happened, the experimenter would know only the most probable member of the contemplated set of subgoals. He would lose a large amount of potentially available information.

The following contract is workable. The subject generally agrees that when he states certain degrees of belief at trial n (regarding the cogency of aiming for certain subgoals), he is really evaluating a probability vector, $\vec{p}(n)$, in which $p_i(n)$ is his belief in the appropriateness of rehearsing the subproblems denoted by $x \in X_i$. Ideally, he would like to throw a dice with its faces biassed by the $p_i(n)$ and to select an alternative, i(M), according to the outcome of the throw.

Now, brains are peculiarly bad at this sort of dice throwing operation. But the automaton is provided with a biassed random event selector[ρ], its bias being the probability estimating vector $\vec{r}(n)$, and its output being a particular value of i(M).

Hence, the automaton can cooperate with the subject insofar as he outputs an $\mathcal{L}^1$ statement, $\vec{r}(n)$ (by adjusting the potentiometers of Fig.8) and commands the selector by submitting $\vec{r}(n)$ to the device.

FOOTNOTE: ρ This is a "variable window width" selection device, the i^{th} "window width" being determined by $r_i(n)$ at the n^{th} trial. Given a specified set of alternatives (set up on relays by the $\mathcal{L}^1$ probability statement of Section 1.5) and given an $\mathcal{L}^1$ probability estimating vector $\vec{r}(n)$, the device accepts an interrogation signal and outputs one member, i, from the set of possibilities. i is selected with probability $r_i(n)$ and is used, like a deterministic statement, to determine the subproblem class for the next trial block.

Clearly, the cooperation is effective only if $\vec{p}(n) = \vec{r}(n)$. But it seems that little practice is needed to acquire the skill of setting $\vec{r}(n) = \vec{p}(n)$ (it will be recalled that the system automatically imposes the constraint that $\sum_i r_i(n) = 1$), and that subjects can do so comfortably after some preliminary experience of working in these conditions. Since deterministic and probabilistic selection are not really distinct (the subject makes a deterministic selection, i(N) as the limiting case of a probabilistic selection), the "event selector" facility supports the "attention directing" contract established in 1.6.4.

1.6.6. Commitments to Goal Directed Performance

The subject might select subgoals in a mercurial fashion unrelated to goal directed activity. In the long run, this is unlikely to happen once that he has agreed to aim for the main goal. But, it may happen at moments when the subject is fatigued or frustrated. To guard against the chance of purely capricious behaviour, the subject is thus required to enter into a commitment whenever he selects a subgoal for the subsequent trial block.

Two sorts of commitment (I) and (II) have been employed, (I) in the pilot experiment and (II) in the main experiments.

(I) 4 lamp stimulus codes are input to the automaton in blocks of 16. Within any one block, each stimulus code occurs at least once, hence the block is "representative". When the subject selects a subgoal, he is required to deal with problem posing stimuli derived (by the automaton) from this sequence of stimulus codes. In doing so, he is required to achieve a minimum level of performance, indicated by a score criterion of $\rho = 15\%$ (where ρ is defined as number of correct responses per block). If he fails to reach this criterion, the next selection is determined by the automaton. Several "automaton procedures" were

devised for selecting i(M), all of them entailing the concept of a difficulty ordering (ABCD is more difficult than any of ABC, ABD,.... is more difficult than any of AB, AC,.....is more difficult than any of A,B....). These procedures will not be discussed because, although the general edict "keep up at least a minimum score" was heeded, and although this type of "commitment" served quite well in the pilot experiments, it was rarely necessary to take control out of the subject's hands.

(II) 4-lamp stimulus codes are input to the automaton in segments of 8 elements punched on a tape. Each tape segment contains one occurrence of each stimulus code. The n^{th} code is read by the automaton and the appropriate stimulus is derived from it and displayed. If the subject's response to the n^{th} stimulus is completely correct ($\sigma_o(n) = 1$), the tape is advanced by one step and the next stimulus displayed in the block consists in the stimulus derived from the $n+1^{th}$ code. If not, ($\sigma_o(n) = o$), the n^{th} code is again read and the n^{th} derived stimulus is displayed. This procedure is flow charted in Fig.12. Hence, although a block consists in 8 first presentations of each stimulus, it may consist in any number of actual stimuli, depending upon whether and how the subject corresponds correctly. Using this scheme, the subject is committed, on entering the block, to getting each response correct at least <u>once</u>. The appropriate score, ρ, becomes the number of <u>first</u> correct responses, i.e. the number of occasions on which the subject is correct when a stimulus is first presented in a block $\Sigma \left[(\sigma(n) = 1) \wedge (x(n) \text{ is FIRST}) \right]_{r \in \text{Block M}}$.

Using this method of obtaining commitment, it is also possible to calculate a very valuable index, namely the number of guessing trials required, as an average, before the subject makes a correct response.[6]

FOOTNOTE: [6] The method is open to two criticisms, namely, (1) that it fails to guard against certain trick responses and (2) that it partitions the set of problems by requiring the subject to concentrate upon one problem until he solves it. So far as (1) is concerned, the chief trick is obvious; working in the complete knowledge of results condition, a subject who fails at the first trial simply inspects the knowledge of results statement and sets his fingers, as an external memory aid, over the appropriate keys. On the next trial, he makes a "spurious" or "copying" correct response. Clearly, this cannot happen in the partial knowledge of results condition. But, even in the complete knowledge of results condition, observation of the subject's behaviour suggests that such a blatent sort of cheating rarely, if ever, occurs. The observation is supported by inspection of the response records for the transiently correct responses, which the trick would produce; they do not appear. To confirm this, some subjects were run in conditions where they had to return their fingers to rest positions away from the response keys between trials (a brief interval being allowed for the transition to and from the response keys). The form and slope of the learning curve did not differ appreciably between the condition <u>with</u> the finger return and the condition <u>without</u> it (the comparison was based upon the same subject learning different rules).

So far as criticism (2) is concerned, the method does impel the subject to concentrate on a single problem and this is quite deliberate. We have also used a much more complicated method, involving a biassed random stimulus selector, similar to the device introduced in 1.6.5. This method does not require the subject to concentrate on one problem at once and it is discussed in Appendix 1.

1.7. The Measurement of Subjective Uncertainty

In either the experimental or tutorial use of the EBPCI method, it is desirable to obtain an index of the subject's uncertainty with respect to what he should do next (I) at each trial and (II) at each trial block end. Of these, (I) is an uncertainty over response alternatives stated in $\mathcal{L}^0$ and is considered in 1.7.1. (II) is the $\mathcal{L}^1$ uncertainty examined in 1.6.5. Further estimation methods are discussed in 1.7.2.

1.7.1. $\mathcal{L}^0$ Response Uncertainty

The obvious expedient would be to determine this uncertainty as a function of the mean (correct) response latencies. Unfortunately, this possibility is precluded (a) because the response interval is restricted to Δt (a relatively minor objection) and (b) because the several components of the response are not independent (and the initial conditions are so contrived that they cannot be independent by the setting assigned to Δt...

However, using the stimulus repetition method of 1.6(6) (II), the subject's uncertainty about responding to a given stimulus (solving a given problem) is reasonably estimated by the number of guessing trials he needs to produce a correct response. Averaged over a block, this index gives a gross measure of uncertainty with respect to the relevant problem subclass.

From the block generation routine, the total number of guesses is clearly BLOCKLENGTH - 8. But, the guessing numbers must be computed with respect to the derived stimuli rather than the 4-lamp stimulus codes of the input tape. Thus, using either subset α or β of Fig.4, the 8 long block code gives 2 distinct derived stimuli for the subclasses X_A, X_B, X_C, X_D; 4 distinct derived stimuli for $X_{AB}, X_{AC}, X_{AD}, X_{BC}, X_{BD}, X_{DC}$; 8 distinct derived stimuli for X_{ABC}, X_{ABD}, X_{ADC} and X_{BDC} and 8 for X_{ABCD}. In view

of this fact, the uncertainty at the n^{th} block is estimated as follows. Let i be the index over the problem classes and let c_i be the number of distinct derived stimuli designating their members. Let $z_j(M)$ be the number of guesses required to obtain a correct response on the first presentation of stimulus, x_j, $j = 1,\ldots c_i$, in trial block M. The required uncertainty at the M^{th} block is:

$$u_i(M) = \frac{1}{c_i} \sum_j z_j(M)$$

Notice that the measure is already in logarithmic form since the number of guesses constitutes the number of branches in a choice tree (not the number of terminal nodes). But there is a tacit assumption that, on average, the same amount of uncertainty is resolved (the same number of equivalent choices are made) at each guess. See also Appendix 8.

This measure is also admissible when the stimuli are derived from subsets in which each possible derived stimulus occurs in each block even though the occurrence frequencies of the derived stimuli are unequal.

1.7.2. $\mathcal{L}^1$ Selection Uncertainty

When these experiments were started, we proposed to employ the subjective probability estimation procedure of Shuford, Massengill and Albert [5][6], who have used their method to estimate the subject's degree of belief in several $\mathcal{L}^0$ response alternatives.[9]

FOOTNOTE: [9] Obviously, the method cannot be applied to the estimation of $\mathcal{L}^0$ uncertainty in the present case since, if the subject could ponder at his leisure (long enough to make a probability statement), there would be no doubt to be resolved. The correct response could be discovered in a manner that is inhibited by the choice of Δt. Once again, we insist that problems <u>exist</u> only insofar as Δt <u>is</u> restricted.

Briefly, Shuford considers a simple $\mathcal{L}^0$ situation in which the subject is presented with a number of response alternatives (say m of them), one being designated as the correct response. He assumes that at any trial the subject has a real subjective probability distribution, $\vec{p}_i(n) i = 1,....m$, in which the $p_i(n)$ are probabilities that the i^{th} response is the correct response. It is also assumed that any statement of these probabilities $\vec{r}_i$ is scored. Now the majority of scoring schemes do not have the property that the subject's mathematical expectation of score is maximised by a veridical statement, i.e. by asserting $\vec{r}_i(n)$ such that $\vec{r}_i(n) = \vec{p}_i(n)$ (in general, the subject is encouraged by the scoring scheme to select the most probable alternative, the largest p_i equal to p_{max}, and to set $r_i = 1$). Shuford has devised a class of game-like scoring system which do have this property, i.e. the mathematical expectation of score is maximised if $\vec{p}_i(n) = \vec{r}_i(n)$.

Initially, it was intended to use a system of this type in connection with subjective probability estimation over the $\mathcal{L}^1$ alternatives. Certain modifications were necessary because, in this case, there is no unique correct response but a distribution of rectitude. Consequently, some mathematical work was undertaken and this is reported in Appendix 2.

However, one of the main features of Shuford's system is that subjects soon learn to use the scoring function and to produce veridical estimates of their certainty, i.e. they learn to set $\vec{r}_i(n) = \vec{p}_i(n)$. Precisely the same result is attained if the subject is provided (as in the technique described in 1.6.5.) with a fair random selector (an electronic dice thrower biassed by $\vec{r}_i(n)$ at the n^{th} trial), which selects the one alternative to be adopted.[o]

FOOTNOTE:

[o] The experimenter cooperates by providing the electronic device to replace the (very ineffective) dice thrower in the subject's brain. In turn, the subject cooperates by providing an input r(n) to this device such that r(n) = p(n).

This, again, is a game-like situation and that is why it works. Because this is so, any mathematically sophisticated scoring function proved unnecessary.

1.8. Discourse Recorded

The whole of the L^1 discourse is recorded either manually or on a punched tape (one or the other depending upon the mechanised response modes employed in the experiment). The L^0 discourse is recorded on a punched paper tape. The data consists in -

(1) Stimulus number for each trial.
(2) Attributes (subproblem classes) selected for attention (recorded redundantly at each trial, since the selection is changed after each trial block, at the most.)
(3) Control information from the automaton (not used in the final analysis of the data).
(4) The rule employed.
(5) Whether or not the entire response is a correct response.
(6) Whether or not the individual response components are correct responses.
(7) The trial block score.
(8) The latency of each response component occurring at a given trial.

This data is either printed out in the format of Fig.14 or the format of Fig.15. The immediate analysis yields data on the existence of response patterns, indicating two response tendencies, namely "stringing" and "grouping" (typical exemplars of these groupings are shown in Fig.16 and Fig.17.) In addition, we compute the scoring function, ρ and the uncertainty measure, $u_1(n)$, discussed earlier.

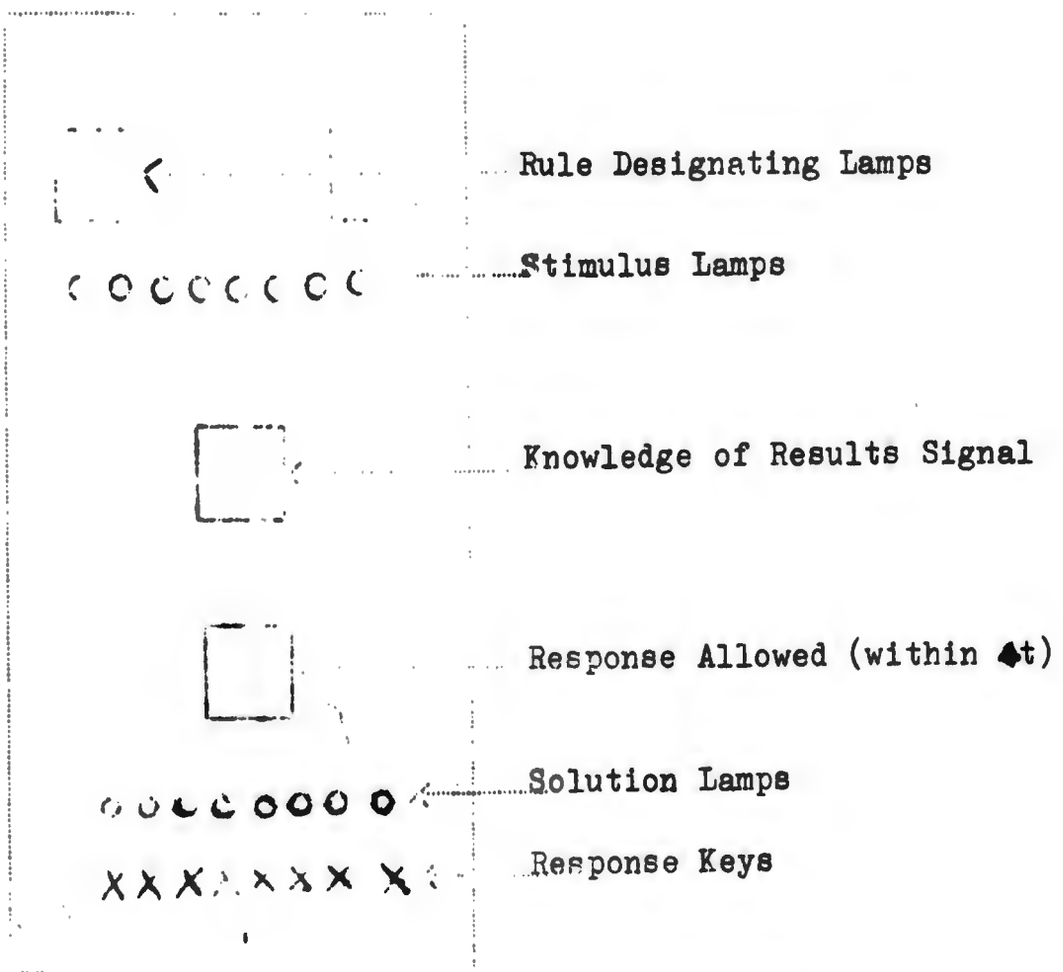


FIG 1. Rule Application Task Display and Response Board. The Subject must respond within an interval of Δt after the stimulus is presented.

FIGURE 2.

Stimulus Numbers	1	2	3	4	5	6	7	8
4-lamp	abch	abdc	bdce	cedf	egdf	fheg	gafh	gahb
3-lamp	abc	bdc	dce	def	efg	fgh	hga	abh
2-lamp	ac	bd	ce	df	eg	fh	ga	hb
1-lamp	a	b	c	d	e	f	g	h

Figure 2. The Original Hierarchy of Stimuli.

FIGURE 3.

	A	B	C	D
Value = 1	a	b	c	d
Value = 0	e	f	g	h

Figure 3. Attribute Values.

FIGURE 4.

Stimulus Numbers.	Stimulus x a	b	c	d	e	f	g	h	Attribute Values. A	B	C	D
1	+	-	-	-	-	+	+	+	1	0	0	0
2	-	+	+	+	+	-	-	-	0	1	1	1
3	-	+	-	-	+	-	+	+	0	1	0	0
4	+	-	+	+	-	+	-	-	1	0	1	1
5	-	-	+	-	+	+	-	+	0	0	1	0
6	+	+	-	+	-	-	+	-	1	1	0	1
7	-	-	-	+	+	+	+	-	0	0	0	1
8	+	+	+	-	-	-	-	+	1	1	1	0
9	+	+	-	-	-	-	+	+	1	1	0	0
10	-	-	+	+	+	+	-	-	0	0	1	1
11	+	-	+	-	-	+	-	+	1	0	1	0
12	-	+	-	+	+	-	+	-	0	1	0	1
13	+	-	-	+	-	+	+	-	1	0	0	1
14	-	+	+	-	+	-	-	+	0	1	1	0
15	-	-	-	-	+	+	+	+	0	0	0	0
16	+	+	+	+	-	-	-	-	1	1	1	1

Figure 4. The Stimuli, Stimulus Numbers and Corresponding Attribute Numbers.

FIG 5. Revised Display and Response Board.

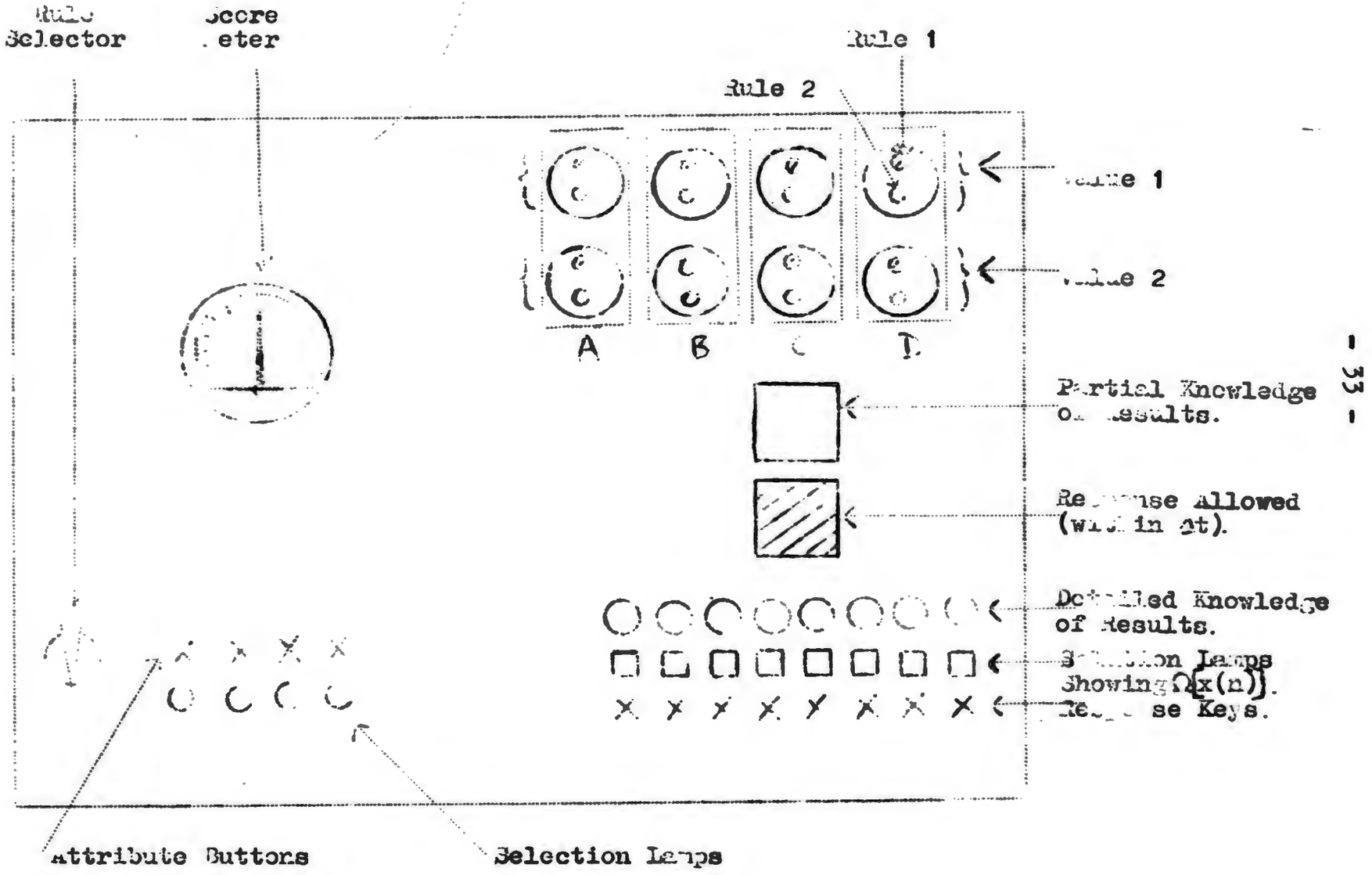
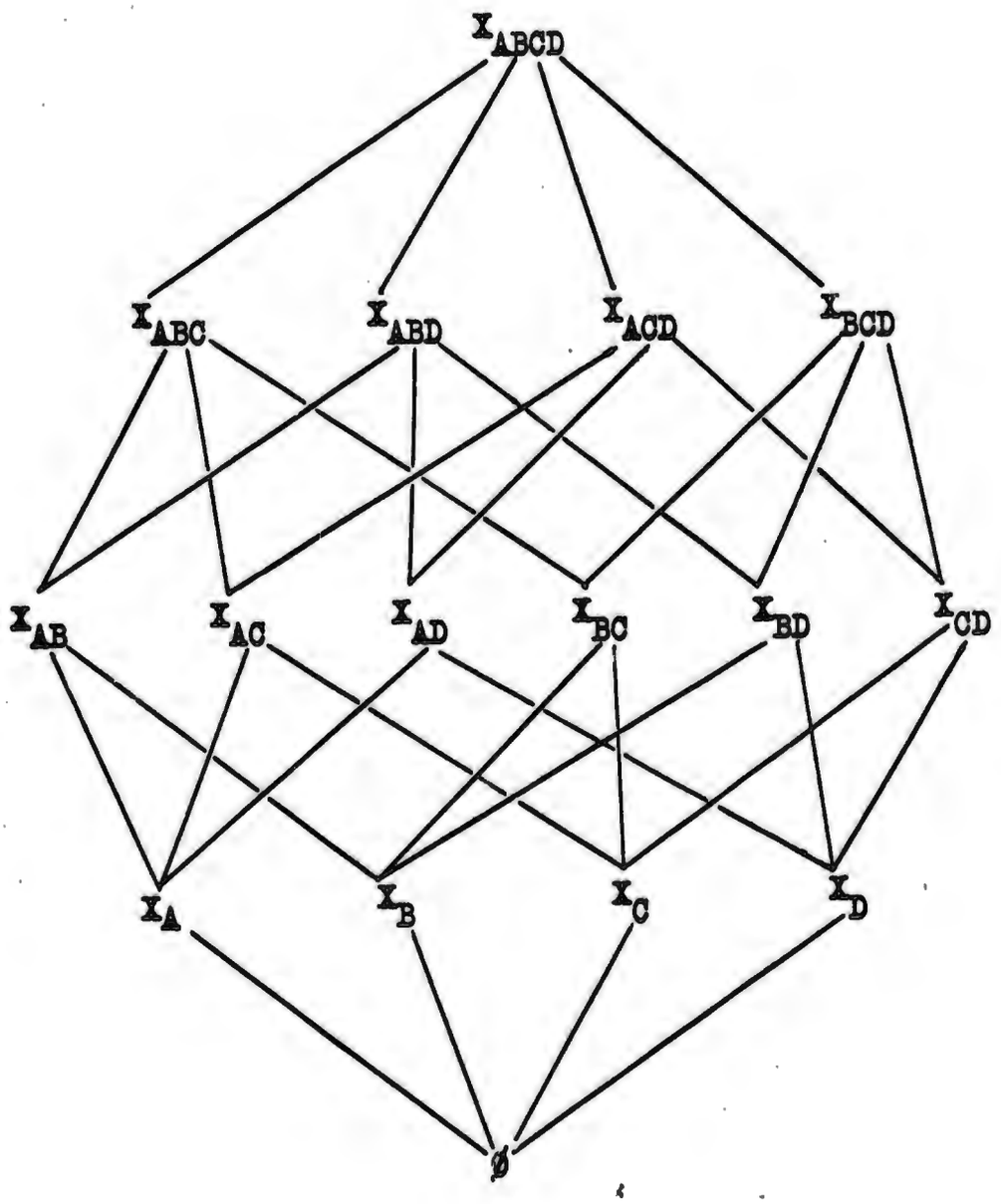


FIG. 6. SUBPROBLEM CLASS HIERARCHY.



X_i, i=ABCD...ABC...AB...A...

FIGURE 7.

Stimulus Number	X_1				X_2						X_3				X_4
	A	B	C	D	AB	AC	AD	BC	BD	CD	ABC	ABD	ACD	BCD	ABCD
1	1 (a)	0 (f)	0 (g)	0 (h)	10 (af)	10 (ag)	10 (ah)	00 (fg)	00 (fh)	00 (gh)	100 (afg)	100 (afh)	100 (agh)	000 (fgh)	1000 (afgh
2	0 (e)	1 (b)	1 (c)	1 (d)	01 (eb)	01 (ec)	01 (ed)	11 (bc)	11 (bd)	11 (cd)	011 (ebc)	011 (efd)	011 (ecd)	111 (bcd)	0111 (ebcd
3	0 (e)	1 (b)	0 (g)	0 (h)	01 (eb)	00 (eg)	00 (eh)	10 (bg)	10 (bh)	00 (gh)	010 (ebg)	010 (ebh)	000 (egh)	100 (bgh)	0100 (ebgh
4	1 (a)	0 (f)	1 (c)	1 (d)	10 (af)	11 (ac)	11 (ad)	01 (fc)	01 (fd)	11 (cd)	101 (afc)	101 (afd)	111 (acd)	011 (fcd)	1011 (afcd
5	0 (e)	0 (f)	1 (c)	0 (h)	00 (ef)	01 (ec)	00 (eh)	01 (fc)	00 (fh)	10 (ch)	001 (efc)	000 (efh)	010 (ech)	010 (fch)	0010 (efch
6	1 (a)	1 (b)	0 (g)	1 (d)	11 (ab)	10 (ag)	11 (ad)	10 (fg)	11 (bd)	01 (gd)	110 (abg)	111 (abd)	101 (agd)	101 (bgd)	1101 (abgd
7	0 (e)	0 (f)	0 (g)	1 (d)	00 (ef)	00 (eg)	01 (ed)	00 (fg)	01 (fd)	01 (gd)	000 (ebg)	001 (efd)	001 (egd)	001 (fgd)	0001 (efgd
8	1 (a)	1 (b)	1 (c)	0 (h)	11 (ab)	11 (ac)	10 (ah)	11 (bc)	10 (bh)	10 (ch)	111 (abc)	110 (abh)	110 (ach)	110 (bch)	1110 (abch

Thus X_A = a,e ; X_B = b,f .....X_{AB} = af, ab, ef, eb ; X_{AC} = ag, ac, e, eg ..... and so on.

Figure 7. Stimulus Subsets X_i Represented in Terms of the Attribute Values.

(continued on next page)

FIGURE 7 (contd.)

Stimulus Number.	X_1 A	B	C	D	X_2 AB	AC	AD	BC	BD	CD	X_3 ABC	ABD	ACD	BCD	X_4 ABCD
9	1 (e)	1 (b)	0 (g)	0 (h)	11 (ab)	10 (ag)	10 (ah)	10 (bg)	10 (bh)	00 (gh)	110 (abg)	110 (abh)	100 (agh)	100 (bgh)	1100 (abgh)
10	0 (e)	0 (f)	1 (c)	1 (d)	00 (ef)	01 (ec)	01 (ed)	01 (fc)	01 (fd)	11 (cd)	001 (efc)	001 (efd)	011 (ecd)	011 (fcd)	0011 (efcd)
11	1 (a)	0 (f)	1 (c)	0 (h)	10 (af)	11 (ac)	10 (ah)	01 (fc)	00 (fh)	10 (ch)	101 (abc)	100 (abh)	110 (ach)	101 (bch)	1010 (abch)
12	0 (e)	1 (b)	0 (g)	1 (d)	01 (ab)	00 (eg)	01 (ed)	10 (bg)	11 (bd)	01 (gd)	010 (ebg)	011 (ebd)	001 (egd)	101 (bgd)	0101 (ebgd)
13	1 (a)	0 (f)	0 (g)	1 (d)	10 (af)	10 (ag)	11 (ad)	00 (fg)	01 (fd)	01 (gd)	100 (afg)	101 (afd)	101 (agd)	001 (fgd)	1001 (afgd)
14	0 (e)	1 (b)	1 (c)	0 (h)	01 (ef)	01 (ec)	00 (eh)	11 (bc)	10 (bh)	10 (ch)	011 (ebc)	010 (ebh)	010 (ech)	110 (bch)	0110 (ebch)
15	0 (é)	0 (f)	0 (g)	0 (h)	00 (ef)	00 (eg)	00 (eh	00 (fg)	00 (fh	00 (gh)	000 (afg)	000 (afh)	000 (agh)	000 (fgh)	0000 (efgh)
16	1 (a)	1 (b)	1 (c)	1 (d)	11 (ab)	11 (ac)	11 (ad)	11 (bc	11 (bd)	11 (cd)	111 (abc)	111 (abd)	111 (acd)	111 (bcd)	1111 (abcd)

- 36 -

Thus X_A = a,e ; X_B = b,f .....X_{AB} = af, ab, ef, eb ; X_{AC} = ag, ac, e, eg ..... and so on.

Figure 7 (co ntd.) Stimulus Subsets X_i Represented in Terms of the Attribute Values.

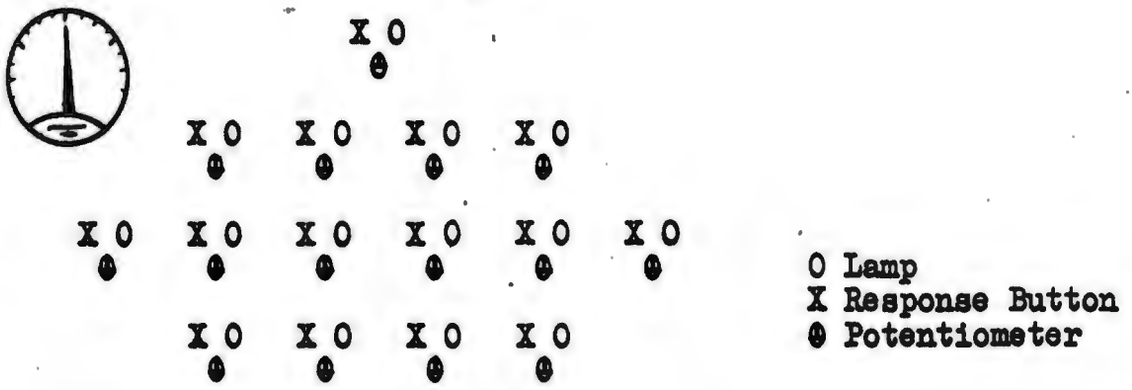


FIG 8 Sub-problem Set Selection Response Board

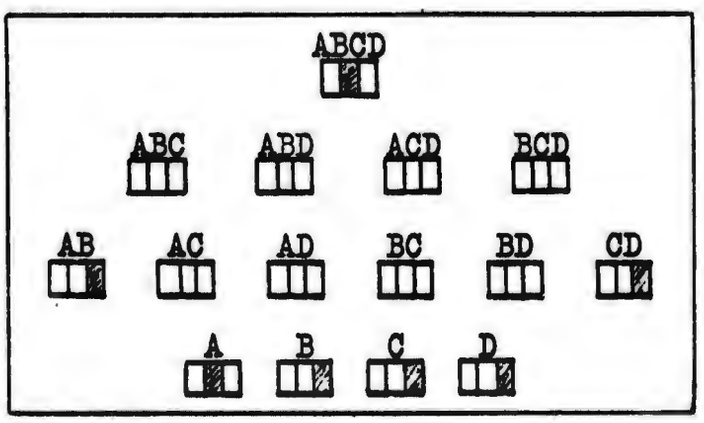


FIG 9 Card For Last Value Of Score

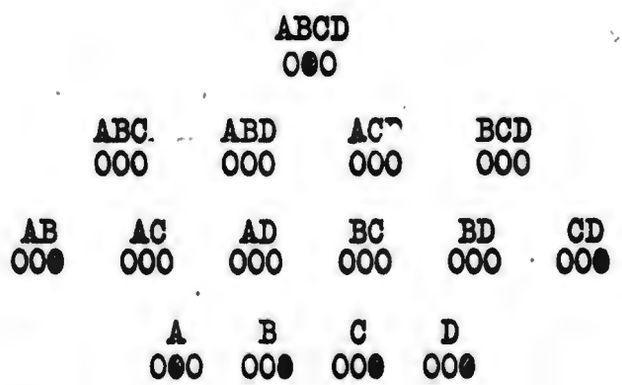
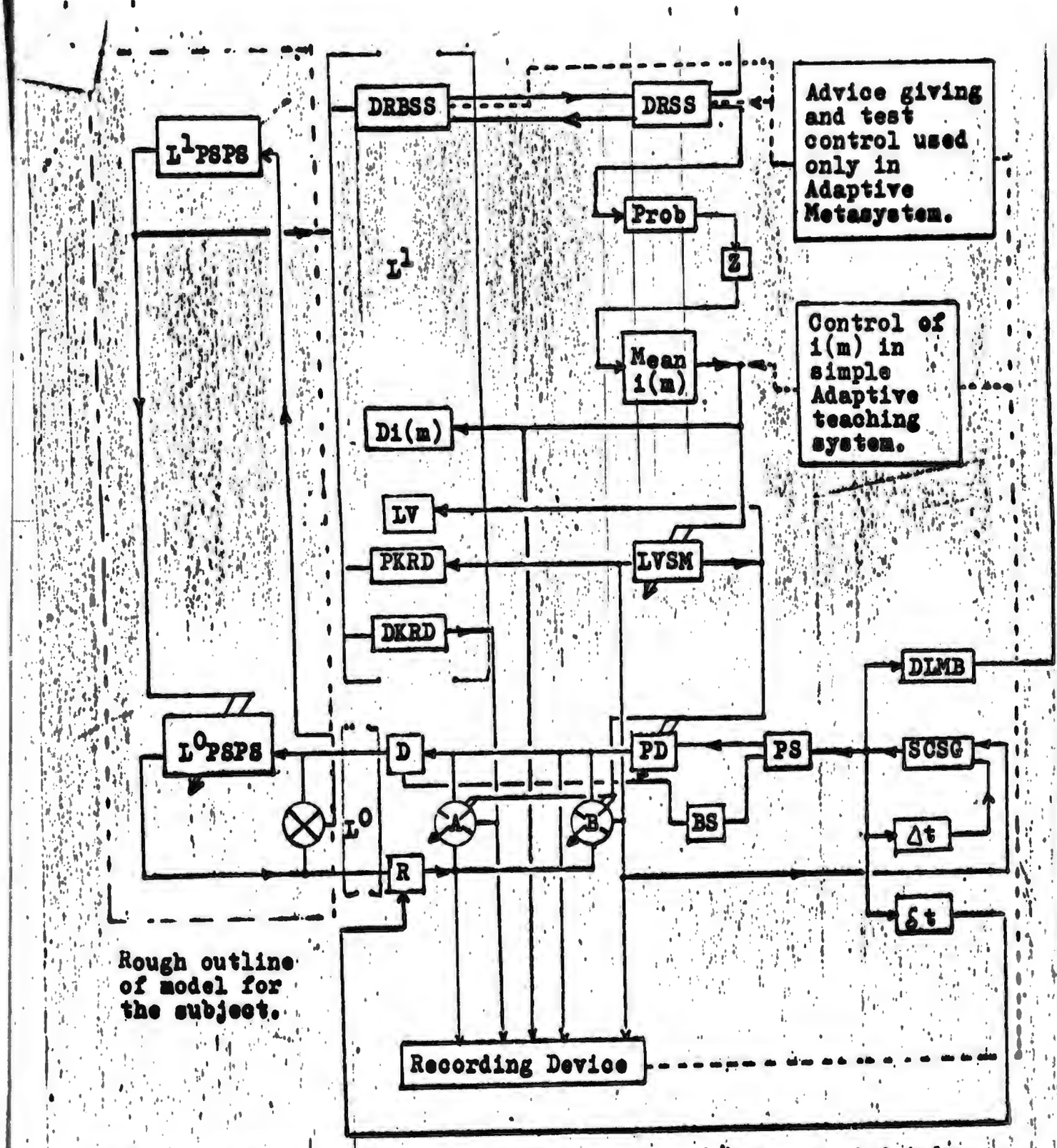


FIG 10 Display For Last Value Or Score



Note 1. In a simple adaptive teaching system i(m) is selected independently of the subject's preference. The subject input is disconnected at Z.

Note 2. In an adaptive metasystem preferences are constrained and tests made using the existing facilities and overriding rather than replacing subject statements.

Figure 11. Automaton for Controlling Experiment in E.B.P.C.I. Method. The figure also shows the additions required if the automaton simulates a simple adaptive machine or the machine in an adaptive metasystem.

see notes overleaf

Figure 11 (contd.)

D.R.B.S.S. = Display and Response board for ℓ^1 selections of subproblems
D.R.S.S. = Demand subproblem class selection mechanism.
Prob. = Probabilistic Selector. Mem i(m) = Register for i(m) value.
L.U.S.M. = Last values score register. Di(m) = Display for i(m).
L.U. = Last value of score display. D.K.R.D. = Partial Knowledge of results. D = ℓ^0 S timulus Display. R = ℓ^0 Response board
A = Detailed knowledge of results comparator. B = Partial knowledge of results comparator. P.D. = Problem Derivation given i(m).
P.S. = Problem sequence of 4-lamp stimuli : given stimulus code selection. S.C.S.G. = S timulus code sequence generator.
D.L.M.B. Detection of last stimulus name in block.
B.S. Background 4-lamp stimulus generator. Δt = Δt time.
δt = δt timer.

In the subject we hypothesise the existence of lower level L^0 problem solving programmes and higher level, L^1, problem solving programmes.

Figure 12.

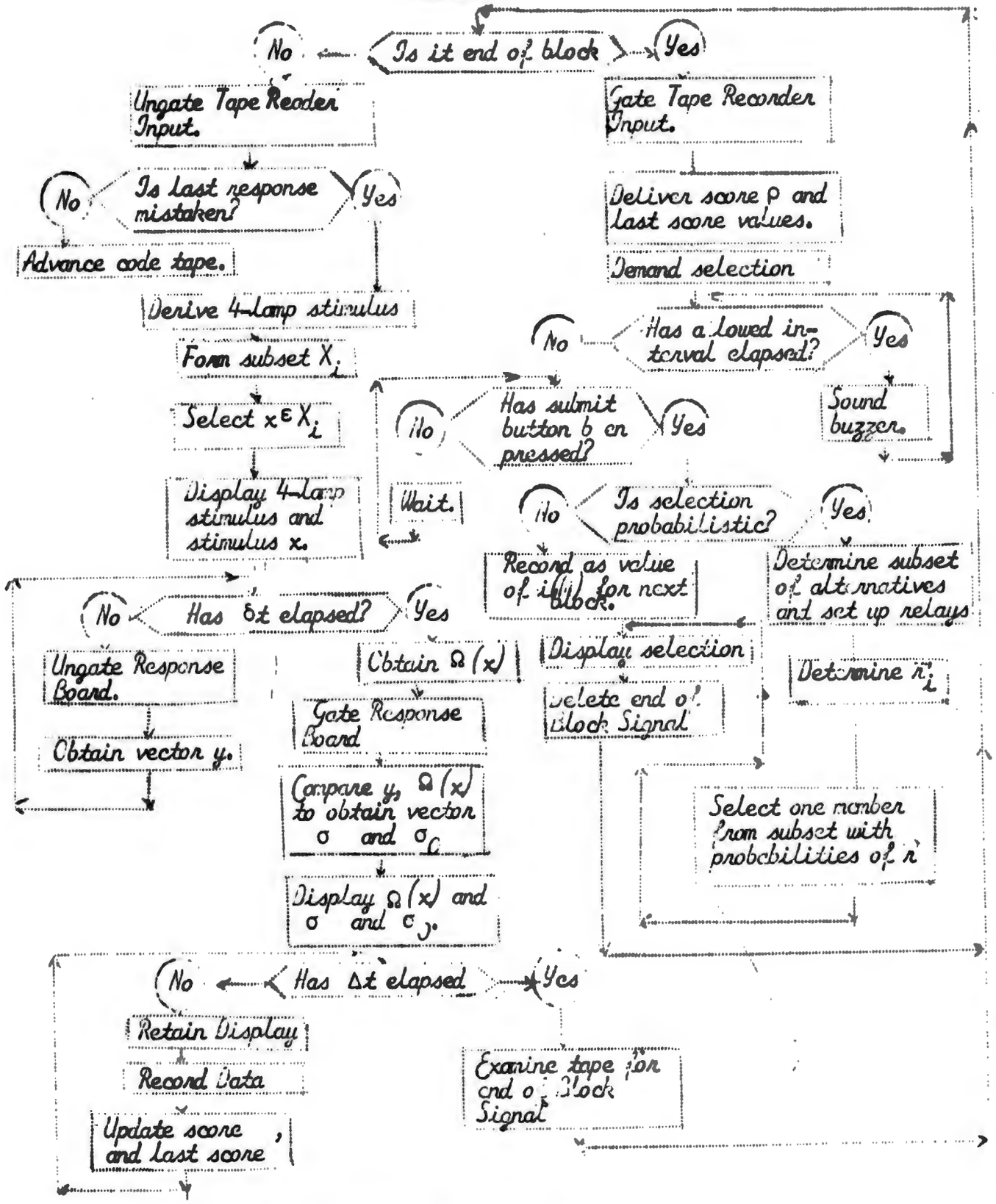


Figure 12. Flow Chart for the Automaton

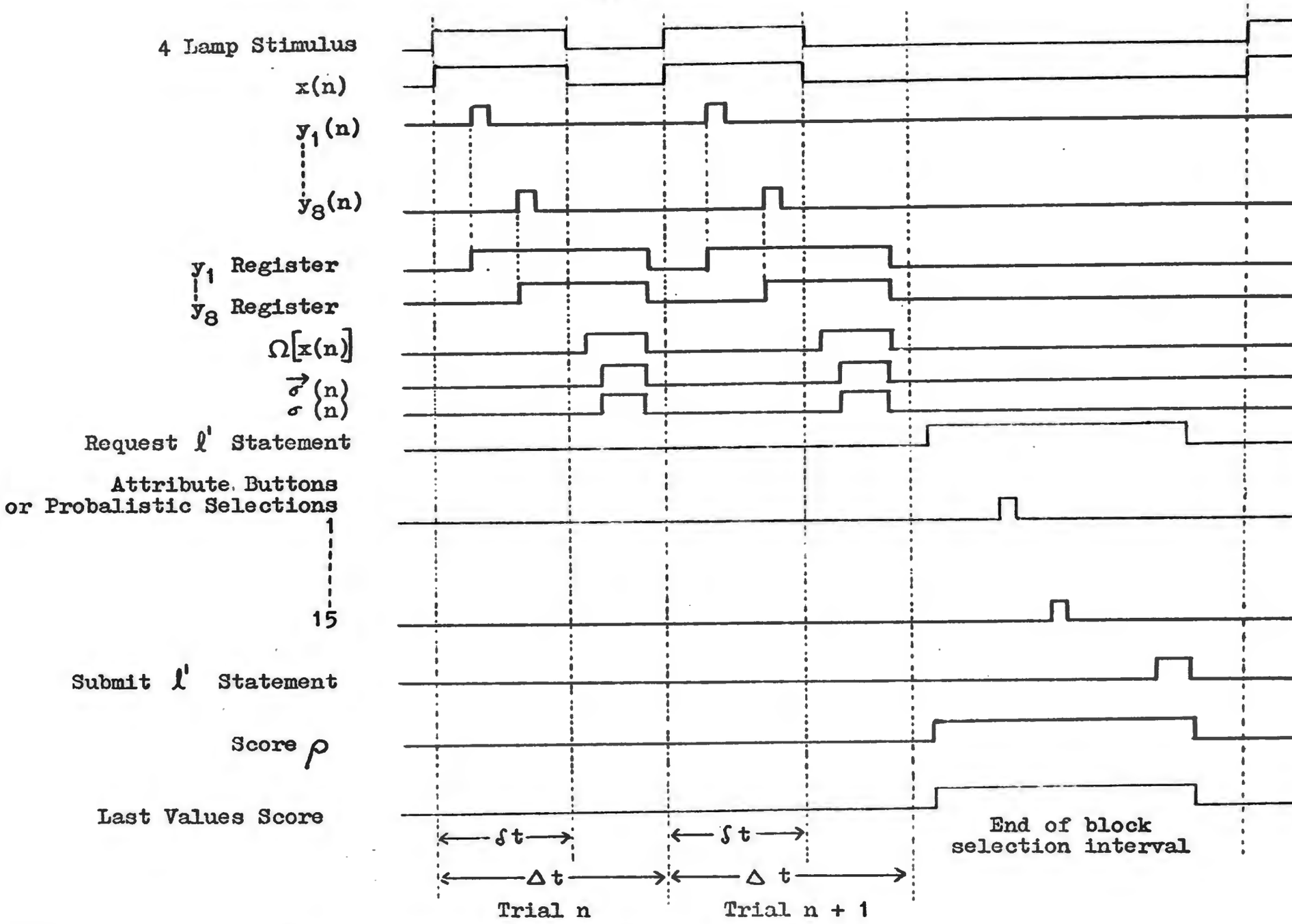


FIG. 13 CYCLE OF EXPERIMENTAL SEQUENCE.

Figure 14.

	Block No.		Attributes Selected		Stimulus presented	Number gives time at which that button presses. 12345678			
*	21	161	R	0011	2	CD	5 7	SS	1
*		162	R	0011	6	GH	-665	DS	0
*		163	R	0011	4	GD	4 4	D	1
*		164	R	0011	3	CD	6 5	SS	1
*		165	R	0011	2	CD	6 3	SS	1
*		166	R	0011	5	GH	53	SS	1
*		167	R	0011	7	GH	43	SS	1
*		168	R	0011	8	CH	* 63	SS	0

1 = all correct
0 = wrong.

Trial No. | Problem No. | Button Nos. | Response Group.

S = Single press
D = Double press

Minus sign indicates wrong response

Asterisk shows which response should have been made.

Figure 14. Computer Format.

Attribute Value Selected		Block End		Any Response		Rule	Latencies for 8 Components of Response
1111	0	0	000	1	1	0	0 1 0 4 0 4 0 1
1111	0	0	011	1	1	0	5 0 0 7 0 7 5 0
1111	0	0	010	1	1	0	0 2 4 0 4 0 0 2
1111	0	0	111	1	0	0	4 0 4 4 0 0 0 0
1111	0	0	111	1	1	0	7 3 0 0 7 7 0 0
1111	0	0	100	1	1	0	0 0 0 0 3 3 3 3
1111	0	0	101	1	1	0	5 1 5 5 0 0 0 0
1111	0	0	110	1	1	0	0 0 4 4 0 0 4 4
1111	0	1	001	1	1	0	1 0 3 0 3 0 7 0
	Spare		Stimulus Number		Correct Response		

Fig. 15. Decoder Format.

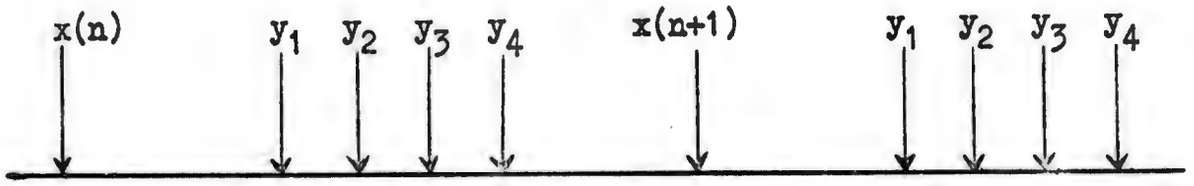


FIG 16. Typical Stringing Responses For Stimulus $x \in X_{ABCD}$

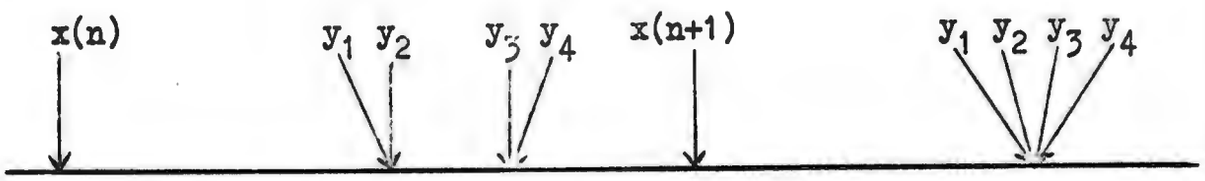


FIG 17. Typical Grouping Responses For Stimulus $x \in X_{ABCD}$

SECTION : 2

EXPERIMENTS PERFORMED AND HYPOTHESES TESTED

2. Experiments Performed and Hypotheses Tested

The experiments involved 42 subjects, each used for between half and one working day. There were two groups of experiments; namely, the pilot experiments, which involved 22 subjects, and the principal experiments, which involved the remaining 20. All of the experiments were conducted within the framework outlined in Section 1, i.e. they entailed interaction in $\mathcal{L} = \mathcal{L}^0, \mathcal{L}^1$, with a controlling automaton and the acquisition of a rule application skill.

2.1. Pilot Experiments

12 subjects were used in the body of this experiment, 6 being assigned to a strong instruction group and 6 to a weak instruction group. The "strong" group were given the goal of dealing specifically with 4 lamp problems and the "weak" group with these and any sub-problems. After a brief practice run (using an irrelevant rule), the subjects learned to perform the rule application task with 4 lamp problems either with no "commitment" constraint (6 subjects) or with the "commitment" constraint of 1.6.6. (I) (6 subjects). After reaching a criterial level of performance ($\rho = 0.75$ at the 4 lamp level), all of the subjects were given a test without knowledge of results and a generalisation test.

6 further subjects were used in a double (or rule alternation task). Their data has not yet been fully analysed and, apart from pointing out that the techniques of Section 1 are, indeed practical in connection with a complex task, we shall not refer to these subjects in the rest of the report.

Finally, 4 subjects were run in forced learning conditions. The automaton was programmed to function as an adaptive teaching machine according to one of the strategies (Plan C), which is described in Section 2.2. There are good reasons for believing that Plan C is a good strategy for learning this skill. Fig.18 is the flow chart

for the teaching procedure. It will be observed that the teaching machine accepts the Plan C as an initial input and that it receives the score ρ and the last score function as an operating input. Its output consists in a subproblem class selection. Hence, it acts as a surrogate for the subject's internal attention directing process.

Throughout all of these experiments, Δt and δt were set at the constant values. $\Delta t = 55$ secs., $\delta t = 3.5$ secs., chosen, on the basis of previous experimental work, so that the subject only has a chance to emit one known response component and to find one response component within the limits of a trial. (There are some indications that Δt and δt may have been too stringently chosen for some of the subjects). Although $\mathcal{L}^1$ interaction took place and the subjects made a subproblem class selection at the end of each trial block, no probabilistic statements were permitted (there is no deep motive behind this restriction; we had not yet instrumented the probabilistic method).

2.1. Numerical Results

The raw data (trials to criterion, T) is shown in Table 1. (Since the stimuli were generated through the procedure of 1.6.6.(II), the number of blocks selected in the course of the experiment is T/8, by definition). There is no significant difference between the strong and the weak instruction group either in terms of trials to criterion (T values) test score or generalisation test score. Further, it was quite apparent that the subjects soon forgot about, or failed to obey, the initial instructions and we conjecture that instructions must be more intimately built into the system, if they are to prove effective. There is a just significant difference (at the 5% level) between the unconstrained group and the group who were run under the commitment procedure of Section 1.6.6.(I). The unconstrained subjects learned the skill in fewer trials than the others because

(from scrutiny of the data) they managed to sustain the laborious process of acquiring some inappropriate subskill (for example, the skill of solving problems ABC or ABCD) by interpolating periods of easy but irrelevant activity (rather than rehearsing the solution of relevant subproblems). We use this result later (Section 2.5.11) in support of a cognitive fixity hypothesis.

Finally, there is a just significant difference (at the 5% level) between the trials to criterion (T values) for the free learning subjects as a whole and those constrained by the adaptive machine; the latter group take fewer trials to learn the skill.

2.1.1. Other Data

The most valuable product of the pilot experiment was the discovery of 3 well defined learning strategies, which, also, appeared to represent quite definite cognitive plans. To some extent, the strategic pattern is obscured by a fairly common tendency to select and stay with some impossibly difficult problem class, usually ABCD. On these grounds, it would be possible to propose a further "strategy" of the form "Try ABCD" or "try A, B, C, D, and go on to ABCD". But learning by either of these expedients is impossible (due to the setting of δt and Δt) and the subjects know it is; further, none of them announced or conceived a strategy of this form (though abortive stretches of behaviour suggest such a thing).

We, thus, prefer to regard these modes of behaviour as tendencies which perturb strategies rather than learning "strategies" in their own right. The tendencies in question can be explained by invoking a cognitive fixity (mooted above and considered in 2.5.11). With this caveat, we distinguish strategies Type A, Type B and Type C.

2.2. Learning Strategies

TYPE A ... A type A strategy is a "chaining" or "sequencing" strategy. It consists in building up the skill of solving 4 lamp problems through competence at solving a nested sequence of relevant and non redundant sub-

problems. The paradigm attention directing sequence is:
A - B - AB - C - ABC - D - ABCD, where the steps of rehearsing B, C or D may be omitted.[o] If the subject runs into difficulty (for example, if he is unable to achieve

FOOTNOTE:

o The paradigms are written as shorthand expressions. Strictly, if i, j, k, l represent any of A, B, C, D, then the chaining paradigm is i - j - ij - k - kjk - l - ijkl = ABCD. Thus, any one of A, B, C or D may be the first member, etc.

sufficient competence at ABC), then he returns to rehearse the solution of one or more constituent subproblems (such as the AB or C subproblems) before continuing to the next stage. This re-entry gambit can be repeated as often as required without contravening the strategy. Thus -

A → B → AB → A → AB → B → AB → C → ABC → C → ABC → ABCD

is the $\mathcal{L}^1$ selection sequence of a Type A strategy and conforms to the paradigm.

TYPE B ... A Type B strategy consists in exploration of the symbolic environment of subproblem classes. The exploration is usually complete. Either the subject has an initial determination to explore or, when he meets with difficulties, he selects a subproblem class he has not yet chosen. Thus, a typical $\mathcal{L}^1$ selection sequence is -

A → B → C → D → AB → BC → AC → BD → ABD → DC → AD → ABC → ABCD

TYPE C ... A Type C strategy amounts to the grouped rehearsal of complementary pairs, which the subject sees as wholes (rather than chains or sequences). The complementary pairs are assembled into a joint chain-like entity to deal with the 4 lamp problem and this, in turn, is later seen and responded to wholistically. The attention directing paradigm[6] is thus -

A → B → AB → C → D → CD ⇄ AB → ABCD

FOOTNOTE: [6] Recall that the paradigm is a shorthand notation. The subject might start with B and D and go on to alternate AC, BD. As it happens, the alphabetic ordering of the attributes impelled the subjects to stick to the literal paradigm for the most part.

where the reverse arrows represent alternate rehearsal of the complementary pair. Certain of the single attribute stages (notably B and D) may be omitted.

There is one variant on this strategy, adopted by two subjects, in which the alternate rehearsal of the specific acquisition of D is omitted and the complementary pairs are replaced by quasi complementary triples -

This will be called the Modified C Strategy.

Type A and Type C strategies may be expected to occur on the basis of the learning model as they are generated by the dominance of one or the other of the main L^1 construction processes. From this, it is possible to infer that the correct response latency patterns (Fig.16 and Fig.17), associated with the Type A strategy, should consist predominantly in string patterns, whereas the correct response latency patterns for the Type C strategy should be predominantly group patterns. These predictions are amply confirmed by the data (discussed in 2.3.4.) and add substance to the strategy classification (in particular, since the correct response latency patterns for the modified C strategy are predominantly groups, there is adequate justification for regarding this strategy as a C variant).

2.3. Principal Experiments

20 subjects were used in the principal experiments; 4 in a preliminary group, run to reveal any major behavioural differences produced by differences in procedure between the pilot and the principal experiments; 10 in a free learning group and 6 in connection with an adaptive metasystem, explicitly designed to teach the skill. Of the free learning group, 5 subjects were assigned to a subgroup receiving complete knowledge of results

(in the sense of 1.5.4. and 1.6.6.), whereas 5 were assigned to a subgroup receiving partial knowledge of results (in the sense of 1.5.4. and 1.6.6.)

All of the subjects in this experiment were run under the commitment procedure of 1.6.6. (II) so that the measure u(n) could be computed as an index of $\mathcal{L}^0$ uncertainty. Further, all of the subjects were encouraged to make probabilistic $\mathcal{L}^1$ statements, $\vec{r}(n)$, as well as deterministic statements.

2.3.1. Design of the Experimental Session

The experiment is laid out in the manner of Fig.18.

(1) There is a brief period in which the value of δt is adaptively varied whilst the subject works on 2 lamp problems with an overlearned one-to-one rule. The criterion sought by the adaptive variation is a value of δt, such that the subject is just able to produce 2 correct response components for 50% of the problems. At least one of the response components is worked out by reference to the rule and the intention is to secure δt conditions in which the subject has 50% chance of doing 2 things (i.e. producing a pair of correct response components) when his knowledge (the overlearned rule) refers to each component separately. Of course, if the period was prolonged, the subject would begin to see the 2 lamp problems as a whole. It is unlikely that this sort of learning did actually take place since the response patterns in the adaptive phase were observed to contain no groups.

The value of δt, thus obtained for an individual subject, was used for that subject throughout the experiment. Further, $\Delta t = \delta t +$ secs, the values of which are shown in Table 2. There is a negative correlation between trials to criterion and the value of Δt, i.e. subjects assigned a short Δt did, on the whole, better than those given a long Δt. Hence, the system is, if anything, under compensated.

This procedure avoids a possible criticism of the pilot experiments, i.e. that Δt may be chosen to make life impossible for some of the subjects. Here, Δt is chosen individually so that the stimuli will pose problems of the requisite sort.

(2) There is a preliminary experiment in which all subjects engage in free learning of a fairly simple rule application skill, the rule being Ω_1 of Fig.20.

(3) After the preliminary experiment is completed, any non metasystem subject is interrogated, using a potentially mechanisable (but not actually mechanised) procedure, which is described fully in Appendix 7. Explicitly the interrogation is a way of questioning the subject, within the cognitive framework of Section 1.5., about (I) the $\mathcal{L}^1$ statements he might make (a sequence of subproblem class selections) and (II) the use he might make of $\mathcal{L}^1$ automaton statements (especially the use of the score and last score values). This pre-experimental interrogation is carried out by the experimenter in an attempt to discern any plans which the subject may have for learning in the main part of the principal experiment. He is told that his announcement of a plan does not commit him to adopting the corresponding strategy if he does not wish to do so and he is also told that he can change his strategy whenever he wishes. All of the subjects did announce a plan of some sort at this stage. The metasystem subjects are not interrogated in this fashion.

(4) In the main experiment the subjects learns a more difficult rule application skill, which involves the rule Ω_2 of Fig. 21 It is conducted in one or other of the following modes; free learning and complete knowledge of results; free learning and partial knowledge of results or adaptive metasystem.

(5) Any non metasystem subject is given a post experimental interrogation using the technique of Appendix 7. However, the experimenter now refers back to actual instances when it is necessary to clarify a point. Metasystem subjects are not interrogated.

Finally, all subjects were asked to make unsolicited comments about their cognition and general strategy. If they did not bring the matter up themselves, they were asked whether they had "visualised" the problems (i.e. "seen" them, or groups of them, in a spatial organisation), or whether they had "verbalised" the problems (i.e. used a mnemonics or "sequential numberings").

In the course of these experiments (both the matching block of subjects and the 10 subjects run in free learning conditions), no new types of strategy emerged and all of the subjects could thus be classified as Type A, B, C. Similarly, all of the behaviour consistent interrogation data was compatible with one or another of these alternatives (with one exception, the 4 Type C subjects announced plans that were inconsistent with the selections they made. This one said that he would try everything, just to be certain about the skill).

2.3.2. Adaptive Metasystem

The adaptive metasystem is a teaching system designed to sustain $\mathcal{L}^1$ as well as $\mathcal{L}^0$ interaction; to eliminate some of the pathologies of learning (especially the espousal and retention of inappropriate strategies); to ensure that a strategy is selected only if it does suit the subject; to ensure that once the strategy is selected, its progressive application is continually controlled, and to foster the status of the subject as a self-organising system when he is coupled to, or attending to, relevant material.

At this point, the metasystem will be specified. Later, after a discussion of the results in Section 2.4., its design will be justified by demonstrating the likely truth of a series of 18

hypothesis in Section 2.5. Superficially, this is a perverse way of presenting matters but it is possibly better to say what a metasystem is and to show that it definitely does work before going on to the details of how and why it works.

The flow chart for the adaptive metasystem is shown in Fig.22. Like the simple adaptive teaching system of Fig.19, it receives an initial input and an operational input. The initial input is a set of strategies (Types A, B and C) in contrast to the single strategy (Type C) of Fig.19. Its operational input consists in $\mathcal{L}^1$ statements by the subject (not used in Fig.19) and two distinct measures of competence, namely ρ and last score (as before) and a measure of the frequency with which different correct response latency patterns are produced. This frequency measure is designated λ and counts the ratio of grouping response patterns to other response patterns, for all completely correct responses. Ideally, the ratio of grouping responses to stringing responses should be computed but the equipment required for doing this was not available when the experiments were carried out.

The output of the adaptive metasystem automaton consists in a sequence of recommendations or choice options that progressively reduce the subject's degree of freedom and may, ultimately, constrain him entirely. Some of the options refer to strategies (i.e. Type A, B or C may be adopted) but, in this case, they are contingent (if you choose a Type A plan, then you must pass Tests A_a, A_b...; if you choose a Type B plan, then you must pass tests B_a, B_b....; if you choose a Type C plan, then you must pass C_a, C_b... and so on). At other points in the training routine, the options refer directly to the selection of different subproblem classes and, once again, they are made contingent upon tests of performance.

The design philosophy of the automaton is that any of the Plans A, B or C might be a "good" one for a particular subject. Further, its "goodness" depends upon the subject's immediate condition and cannot be reliably predicted, except by asking him. However, it is also true (1) that any plan he chooses must be made explicit (so that the subject can anticipate the consequences of adopting it) and (2) that its goodness for this subject depends upon his possessing the mental equipment to execute the corresponding strategy. To some extent, his possession of this machinery is sufficiently indicated (given a suitable test) by "Score", ρ, and "Last Score". But there are other factors, notably the ability to group and/or to make stringing responses, of which the subject is normally unconscious. The automaton is designed to bring these factors into consciousness by specifically instructing the subject (who has chosen a given strategy) to perform tests in an attempt to group ("see things as a whole") or string ("chain items together") and, after the test is completed, by presenting him with the relevant data. The object of all this is to ensure that the subject does adopt some strategy but does not get into a condition of cognitive fixity[o] by adopting an inappropriate strategy.

The design philosophy also incorporates the idea that a subject should be allowed as much freedom of choice as is compatible with the prosecution of the chosen strategy. Thus, no strategy is restrictive in respect to the rehearsal of A, B, C, D on their own and here complete freedom is permitted (at this stage the subject need not even commit himself to a Plan). Soon, however, restrictions are imposed by a Type B or a Type C strategy. The Type B strategy permits any selection of a subproblem class. But, in view of the fact that exploration fills intermediate memory with redundant operators[oo], the performance criterion, which must be satisfied by any selection, is very high indeed (the subject is not allowed to fill his intermediate memory with mistaken operators).

FOOTNOTES:

o As in 2.5.10.

oo As in 2.5.5. and 2.5.6.

Given these preliminary remarks, the detailed operation of the adaptive metasystem is made explicit in Fig.22. It should however, be emphasised that a metasystem, which maintains interaction in $\mathcal{L}^1$ as well as $\mathcal{L}^0$, is a genuinely conversational system.

2.4. Gross Results

The gross data (in terms of T, trials to reach an 80% criterial of proficiency* over 2 blocks of 4 lamp problems) appears in Table 3. The value of T_1, is the trials to criterion score for the preliminary experiment (item (2) of Section 2.3.1.) and T_2 the score for the main experiment (item (4) of Section 2.3.1.) The number of blocks to reach criterion, E, (i.e. the number of distinct $\mathcal{L}^1$ selections) is given, for completeness, in Table 4 (though this data is not used immediately). Table 5 shows values of the ratio -

$$T^* = \frac{T_1}{T_2} .100\%$$

The preliminary experiment is easier to perform than the main experiment so that, generally, this ratio is less than 100% and it will be low valued insofar as the main task was hard to learn relative to the standard preliminary task. If any factor, such as the employment of a metasystem, improves learning efficiency in the main experiment, then T* will increase. Thus, T* is a relative efficiency measure and, since the preliminary task is used as a control task, its within group values are unaffected by the initial setting of Δt (which will influence both the T_1 and T_2 values in the same sense).**

FOOTNOTES:

*From 1.6.6. (II) the value of ρ is computed as the number of first correct responses in a block, which is generated by 8 distinct stimuli but is of indefinite length.

**In fact, as in Table 2, the Δt values are freely scattered. No significant difference between Δt (free learning) and Δt (adaptive metasystem), and we claim, in any case, that adjustment of Δt serves to standardise the task. But, whatever view may be taken in this matter, between group comparisons of T* factor out the effect of Δt variation.

The primary hypothesis is that learning in an adaptive metasystem is more effective than free learning in the same conditions. To show this, the mean T_2 values for the free learning group are compared with the mean T_2 values for the adaptive metasystem group. The difference is significant (T_2(free learning) > T_2(adaptive metasystem)) at the 0.1% level, $0.001 > p$, which strongly supports the hypothesis. To make the point in a manner that is independent of Δt (and which factors out some of the wide individual differences), the mean T* values for the adaptive metasystem group are compared with the mean T* values for the free learning group. This difference is also significant (T* (Adaptive Metasystem) > T*(Free Learning)) at the 0.1% level, $0.0001 > p$.

It will be observed that some of the free learning subjects have low T values and, thus, learned efficiently. The hypothesis is that these subjects have a mind which incorporates an effective "teaching device" (i.e. that their attention directing mechanism acts like an adaptive metasystem). The adaptive metasystem is, thus, expected to bring <u>all</u> T values down to the level achieved by these gifted people (to bring all efficiences <u>up</u> to this level). Inspection of the raw data supports this point of view.

Comparison of the mean T_2 (Adaptive Metasystem) with the mean T (simple adaptive system) reveals a difference, which is significant at the 0.5% level (the Metasystem being more efficient than the simple adaptive device using Plan C). Some of this difference is spurious since it reflects (1) the influence of the "commitment" procedure of 1.6.6. (II) as against the commitment procedure of 1.6.6. (I) (or no commitment procedure at all) and (2) the effect of probabilistic $\mathcal{L}^1$ interaction. Thus, Mean T (Pilot Experiment) is greater than mean T (Principal Experiment, Free Learning Subjects) and this difference is just significant (at the 5% level).

It is almost certainly possible to demonstrate a highly significant superiority for the adaptive metasystem as compared with the simple adaptive system (using Plan C), when the underlying stimulus generation procedure is given by 1.6.6.(II) in both cases. But some caution is needed in this matter. It is likely that almost any subject can be taught quite effectively using Plan C. The adaptive metasystem has the effect of selecting Plan A, B or C to suit the individual (and for 3 out of 6 subjects, Plan C was follows). Some of the superiority of the adaptive metasystem will show up as a difference between "Optimal Plan Teaching, other than Plan C". On the other hand, some of it will show up as a difference attributable to the motivating effect of engaging the subject (with the adaptive metasystem) in conversational interaction. These two facets of the situation are, of course, related since a conversation could hardly be set up in the absence of strategic alternatives to discuss.

2.5. Salient Hypotheses and Supporting Data

Within the general framework afforded by the learning and teaching model described in Section 5, it is possible to make various predictions about the form and conduct of learning in the systems described in Section 1 and, up to this point, in Section 2. Most of the following hypotheses refer primarily to the functioning of an adaptive metasystem. Indeed, their plausibility is the underpinning required for a cogent theory of how the system works. Some of them have a more general significance in the field of (Cybernetically oriented) psychology. The few subsections marked ※ have no direct bearing upon the question of adaptive metasystems and deal with interesting hypothesis that happen to be supported by the present body of data.

The subsections (apart from those marked by ※) are arranged in a logical order insofar as they present the hypotheses that must be adopted in developing a theory. Some of them are supported by statistically significant data. Some can be more appropriately supported by evidence akin to a clinicians. For some, no more than a discursive justification can be provided.

2.5.1. The Integrity of Response Patterns

A regularity in the empirical correct response latency patterns has already been noted. Its form corresponds, quite accurately, to the output of the learning model (which is used to represent the student in the oretical studies). Within this model, responses are produced by the application of "problem solving programmes". These are of several different types, namely (a) simple operators which, on application, select just one response component (b) Strings of operators (formed by an L^1 construction process called "concatenation"). When applied, an operator string produces a regularly ordered sequence (or string) of response components, in the sense of Fig.16 and Fig.17 (c) Complex operators, which are applied to a complex domain and produce several response components in parallel, i.e. a "group" in the sense of Fig.16 and Fig.17.

It is maintained that the "problem solving programmes" that lead to correct responses are stable entities. Hence, in circumstances where the subject may be expected to have only one method of dealing with a problem at his disposal, the observable (real life) response patterns should, also, be stable. The "circumstances" required for uniqueness are that the subject adopts a substantially non redundant learning strategy, or has this imposed upon him. The Type A strategy and the Type C strategy are non redundant (the modified Type C is probably non redundant also). On the other hand, the Type B strategy is, in general, highly redundant and the intermediate memory of a subject who adopts it may be expected to contain various different programmes for solving the same problem. Hence, in the case of a Type A or a Type C strategist, the correct response latency patterns should have an integrity of the following sort.

Consider a stimulus, x_0, designating a problem in a 2 attribute class (the least problem that will give rise to a non trivial pattern in the response). The possible correct response patterns for the response y_0 to x_0 are -

$$\left\{\text{Latency } (y_i) = \text{Latency } (y_j)\right\} \text{ (a group)}$$

$$\left\{\begin{matrix}\text{Latency } (y_i) > \text{Latency } (y_j) \\ \text{Latency } (y_j) > \text{Latency } (y_i)\end{matrix}\right\} \text{ (two different strings)}$$

Let us call any one of these Patterns "P". Similar comments apply to n tuple responses for $n > 2$ but, as in Fig.16 and Fig.17, the number of possible patterns, P, is much greater.

Now, integrity involves the following condition. If a pattern, P_0, characterises the correct response, $y_0(n)$, to a stimulus $x_0(n)$, displayed at trial n, then, if x_0 recurs at some later trial, n + m, either (1) $y_{n+m}(n) = y_0(n)$ should be correct and should have pattern P_0 or (2) $y_{n+m}(n)$ should be mistaken (in which case no definite prediction can be made). Clearly, if P_0 maintains its integrity in the context of x_0 and y_0 over several repetitions, the evidence in favour of its integrity is enhanced and the integrity scores (obtained by counting events of the type cited above) might be weighted proportionately, though this expedient has not been adopted.

To test the integrity hypothesis, counts of P_0 preserving recurrences have been obtained over the Type A and C strategists amongst the subjects using the adaptive metasystem. A rough count over the free learning subjects yields a similar picture. The data is presented in Table 6 and supports the view that correct response patterns are particulate and stable (thus, indirectly, justifying their identification with the isomorphic output of the learning model).*

FOOTNOTE: *The same sort of analysis can be carried out in respect to the production of complex patterns. This question is taken up in Appendix 6.

Clearly, data from Type B subjects cannot be examined in the same light, since a priori these subjects may be expected to have several equivalent problem solving programmes for at least some of the problems; hence, correct responses that do not preserve P_0 would not indicate the instability of these programmes (it would simply imply their diversity).

2.5.2. The Hypothesis of Definite Strategies

Subjects should adopt definite learning strategies. It has already been argued that they do so. The frequency of occurrence of each type of strategy A, B, C and Modified C, is 3, 3, 2, 1 in the principal free learning experiments.

Appendix 3 shows the raw data gleaned from the 10 subjects used in the free learning condition of the principal experiment (both the preliminary run and the main run). Appendix 4 contains similar data from the 6 principal experiment subjects employed in the adaptive metasystem. With one exception (namely subject 11), all of the subjects could be assigned to one of the strategic types. The ambiguity with respect to subject 11 is resolved by the knowledge (his statement) that he changed his strategy in the course of the experiment. Only one subject (namely 9) is assigned to the Modified C category (which may be viewed as slightly arbitrary).

The aptness of the strategic classification is chiefly supported by the possibility of assigning subjects to strategic types and by the coherence of the sequential matching procedure cited in 2.5.4., below. However, certain gross indices also indicate a difference between Type A, B and C. Figs. 23 and 24 show histograms of the subproblem classes selected by the free learning subjects (Fig.23 with the "ambiguous" subjects, 9 and 11, Fig.24 excluding them). There is a significant difference between the numbers of different subproblem classes selected, as shown in Table 7.

2.5.3. Strategy Type Predictability Hypothesis

If the strategy types are realistic, it should be possible to predict them from the learning model (vice versa, if they can be predicted, this tends to verify the model).

Such a prediction is certainly possible in the case of Type A and Type C. The Type A strategy is the best attention directing strategy in a model so biassed that the $\mathcal{L}^1$ concatenation process is predominant (so that, as in Section 5, the model is prone to build up problem solving programmes that are operator strings). Conversely, the Type C strategy is the best attention directing strategy for a model so biassed that $\mathcal{L}^1$ substitution (Section 5) is the predominant process (as a result of which any operator string is substituted, as rapidly as possible, by a complex operator). The modified Type C strategy is a variant that can be produced within the model by an appropriate adjustment of its parameters, but it makes less effective use of the intermediate memory.

The Type B strategy is more difficult to predict. The model is provided with a certain "need to learn" and, in excess, this would generate an exploratory behaviour. But the exploratory behaviour makes very inefficient (highly redundant) use of the intermediate memory and might be positively harmful if the stored problem solving procedures are mistaken as well as redundant. As a practical recommendation, it is, thus, proposed that exploration is permissible if, and only if, it is highly successful.

2.5.4. Consistency Hypothesis

If a subject adopts a particular strategy, then his behaviour should be consistent with this strategy in the sense that the frequency of correct response latency patterns should follow a predictable trend, namely -

% Groups (Type C) > % Groups (Type B) > % Groups (Type A)

The data shown in Table 8 amply supports this hypothesis.

2.5.5. Description of Strategies as Plans

Generally speaking, a subject should be able to describe his plans in a suitable interrogation language as well as using them prescriptively in his $\mathcal{L}^1$ subclass selections. Hence, a correlation may be anticipated between the plan descriptions given before the main part of the principal experiment (using the interrogation language for talking about $\mathcal{L}^1$ statements), the plans themselves and the post-experimental descriptions.

This correlation exists for all subjects in the principal experiment, who adopted a plan of Type A or Type C but, apart from one subject who stated his intention to explore the environment exhaustively, it does not hold for Type B (exploration).

The supporting data is provided by a sequence similarity measure (Appendix 5), values of which are shown in Tables 9, 10 and 11. There is likely to be a correlation between the actual strategy and the post-experimental description, just because of recollection. Hence, it is not surprising that the B strategists in Table 9 do (to some extent) recall what they were up to. The important difference between the "B's" and the "A's" or "C's" shows up in Table 10 and Table 11.

2.5.6. Evolutionary Character of Type B Strategy

From the lack of pre-experimental, actual, correlation in the case of most Type B strategists, we infer that a Type B strategy usually evolves in the course of learning, rather than being pre-determined. There is some circumstantial evidence to support this point of view, but the data is insufficient to justify any firm statement.

2.5.7. Equivalence of Strategies

So far as the learning model is concerned, it is an open question whether one sort of strategy should be more successful than another (i.e. should lead to more efficient learning). True, a Type B strategy looks pretty inept. But the fact is, this strategy (which is ideal for producing an intermediate memory full of redundant problem solving programmes) would be well suited to a model with a defective $\mathcal{L}^1$ application procedure (Section 5). For Type A and Type C, it has already been argued (Section 2.5.3.) that A is good for one plausibly biassed processing mechanism and C for another.

The question is, "do real subjects have a diversity of mental organisations equivalent to different biassed processing mechanisms" (when Type A, B and C should be effective for different individuals), or "do they have just one mental organisation" (when there is likely to be one "most successful" strategy).*

To investigate the matter, the T-values are compared for each type of strategist in the free learning groups of the principal and pilot experiments. If all of the strategies can be effective, no significant difference would be expected; if not, one strategy should be markedly superior to the others. In fact, the differences are not significant ($p > 0.07$) and it is possible to find subjects who learn successfully under each strategy (as in the footnote, it is this fact, which, from a purely pragmatic point of view, leads to the recommendation of an adaptive metasystem rather than a simple adaptive system as the appropriate teaching instrument for this skill).

FOOTNOTE: *If one plan is better than the others and if this plan is independent of performance, (i.e. it entails no specific rehearsal routines) then effective teaching is clearly a pure feedforward process. Either the subject should be instructed to adopt this plan or he should be placed in a rigid training routine, which prescribes the plan on his behalf. This does not exclude knowledge of results, for a knowledge of results signal is no more than feedback to the subject, provided by an external comparator. It is not a feedback control of the subject, in which his performance (or some other aspect of his behaviour) determines the next move in the teaching process. Thus, a linear teaching machine programme is a feedforward routine.

If one plan is better than the others, but it is dependent upon performance, effective teaching is structurally a feedforward process in which recapitulation and, or, pacing are controlled by local feedback. Perhaps a minimal example of such a routine is a remedially branching programme (though the simplest of our own adaptive teaching systems fall into the same category).

If all of the plans are effective for some subjects (or for the same subject in a different mood, with different motivation, or with a different attitude), then the teaching system must be more complex. In general, it must enquire what plan the subject wishes to use and it must allow him to use it insofar as he passes certain tests of ability and suitability. Teaching systems of this type are adaptive metasystems since $\mathcal{L}'$ as well as $\mathcal{L}^0$ discourse takes place between the controller and the subject.

2.5.8. Probabilistic ℓ^1 Statements

When a subject makes a probabilistic ℓ^1 statement designating some set of subproblem classes for attention, it is predicted that he will keep those members of the set not actually selected, in mind. Thus, if he asserts belief in the cogency of examining $X = (X_1 X_2 \ldots)$ at the n^{th} trial, and the automaton selects $X_o \in X$ at this trial, then the subject may be expected to select some members of $X - (X_o)$ in a ℓ^1 statement at trial $n + m$. On inspection of the records in Appendix 3, 78% of the ℓ^1 probabilistic statements have this property and many of the remaining 22% can be ascribed to the specific strategy change of subject 11.[*]

2.5.9. Mental Tautomerism

When a subject is alternately rehearsing a pair of sub-skills (for example, in A Type C strategy, when he directs his attention to subproblems AB at one block and DC at the next), there should be evidence of a "mental tautomerism". Specifically, the subject may be expected to make ℓ^1 probabilistic statements, which indicate that he is contemplating the set $X_{AB} \cup X_{DC}$. This hypothesis is not supported by the behaviour of most Type C strategists, but the single subject (Appendix 4, Subject 11), who uses the modified Type C strategy, does exhibit tautomerism very markedly, indeed.

2.5.10. The Hypothesis of Cognitive Fixity for Strategies

If the subject has invested a lot of effort in building up and storing the problem solving programmes required to execute a plan, then, by hypothesis, he will be loath to relinquish this plan, even though it is demonstrably ineffective in his own case. If it exists, this disposition will be manifest as a cognitive fixity not unlike cognitive dissonance.

FOOTNOTE:

[*] If, as proposed, the probabilistic statements do reflect the subject's anticipation, then a practical adaptive machine can use them in keeping track of his detailed planning.

Cognitive fixity is suggested by the invariance of the initially ~ strategy, whether it is successful or not. All but two subjects (namely 6 and 11 in Appendix 4)[ø] showed a tendency to carry the plan adopted spontaneously in the preliminary experiment over into the main experiment. The similarity indices (Appendix 5) of the preliminary and the main plans are shown in Table 12. There is no significant correlation between these figures and the trials to criterion in the preliminary experiment, thus indicating that the preservation of a plan is independent of its success. It should be emphasised that all of the subjects were allowed and even encouraged to modify their plans both before and after the pre-experimental interrogation. As a further point, only one subject really changed his plan in the course of the experiment.

2.5.11. Cognitive Fixity for Parts of a Strategy or Specific Selections.

A further hypothesis is that cognitive fixity should be manifest in relation to particular subproblem class selections, a prediction which stems from the learning model. In order to make the model learn about problems of moderate difficulty, it must be equipped with a mechanism that puts a premium on the set of programmes constructed (in intermediate memory) within the context of a particular subskill[øø]. It must evaluate its previous efforts at programme building and tend to remain on the

---

FOOTNOTES:

[ø] They "showed no tendency" simply because they "gave almost no information", i.e. they rehearsed ABCD alone in the preliminary experiment.

[øø] This may also be interpreted as a mechanism that makes the learning model strive after partially achieved, but so far, incompletely achieved, subgoals.

job. Such a mechanism is notoriously liable to "overshoot", so that the learning model is easily trapped in a state of fruitless endeavour, i.e. in "cognitive fixity".

Effects of this type were noticeable in the pilot experiment and were noted in Section 2.1.1. as typical of subjects constrained by the commitment procedure of Section 1.6.6.(I). Similar effects occur in the principal experiments (referring to Appendix 3), the behaviour of Subject 5, trials 19-252 and 292-430; Subject 6, trials 245-315 and 402-437; Subject 8, trials 42-124 and most of the ABCD performance; Subject 12, the ABCD marathon starting at trial 409; Subject 13, part of the BC performance from trial 100, part of the ABC performance from trial 249 and most of the ABCD performance starting at trial 450).

The adaptive metasystem records in Appendix 4 obviously reveal no symptoms of this type since the adaptive metasystem is designed to avoid cognitive fixity with respect to inappropriate plans or inappropriate subproblem class selections. It is conjectured that much of the improvement in learning is due to this restriction.

2.5.12. Complete Knowledge of Results and Rate of Learning

The problem solving programmes in the learning model are so organised that a great deal of computational effort must be expended in order to make use of complete (as against partial) knowledge of results during its normal operation. This, incidentally, is a consequence of the logic of the system and is not a matter of programming as such. Thus, in normal operation, it is more "profitable" (and, hence, more "usual") for the learning model to delete a problem solving programme on the basis of falsifying partial knowledge of results information than it is for the model to remedy a defective problem solving programme by the proper employment of complete knowledge of results. This statement, whilst true for normal operation, is not true on those occasions when the learning model

makes extensive use of its guessing routine. In general , it will guess when it is in difficulties, i.e. when it has no applicable problem solving programmes available and when it is unable to construct them in any other way.

It is, thus, predicted that the provision of complete knowledge of results will have a minimal influence upon the gross index of trials to criterion, T. This prediction is confirmed by a comparison (Table 3) of Mean T (Complete) with Mean T (Partial), there being no significant difference between these values.

2.5.13 Complete Knowledge of Results and Block Length

Using the commitment procedure of Section 1.6.6.(II), the length of a block is a gross index of l^{o} uncertainty. By hypothesis, complete knowledge of results will be used (if it is provided) in "uncertain" conditions (because, in these circumstances, the model is impelled to make guesses). Hence, there should be a shorter mean block length for subjects run in the complete knowledge of results condition than there is for the group run with partial knowledge of results. This prediction is confirmed by comparing Mean E (Complete) with Mean E (Partial), using the data in Table 4. The difference is in the expected sense at the 3% level of significance, $0.03>$.

2.5.14 The Adaptive Metasystem and the Subject's Experience of l^{o} Uncertainty

Since the adaptive metasystem is designed to keep the subject's uncertainty within limits that allow him to learn ($H_1 > H > H_2 > 0$, as in Section 3), it is predicted that the l^{o} uncertainty will be considerably less in the adaptive metasystem than it is in free learning conditions. This hypothesis is tested by comparing Mean E (Adaptive Metasystem) with Mean E (Free Learning), using the data from Table 4. The difference is in the predicted sense at the 0.1% level, $0.001>$.

Further, the difference between the values of Mean E (Complete Knowledge of Results Condition) and Mean E (Adaptive Metasystem) is significant at the 0.2% level, 0.002 >p. Using the Jonckheares Trend Test, the trend E(Partial) E(Complete) E(Adaptive Metasystem) is significant at the 0.1% level, 0.001 > p.

2.5.15. A More Accurate ℓ^0 Uncertainty Measure

The variable u(n) of Section 1.7. provides a more accurate index of ℓ^0 uncertainty than block length. Table 13 shows the mean value of u for all subjects and there is a very marked difference between the free learning subjects (greater) and the adaptive metasystem subjects (less). This difference is significant at the 0.1% level, 0.001 > p.

2.5.16. Detailed Examination of Adaptive Metasystem

One important influence of the adaptive metasystem is to reduce the height and length of the ℓ^0 uncertainty peaks that generally appear when the subject tackles difficult problems. As mentioned before, the metasystem also eliminates that form of cognitive fixity which impels the subject to aim prematurely and unproductively for the overall goal (rehearsing the solution to x X_{ABCD}). Both effects are demonstrated by a comparison between Fig.25 (Free Learning) and Fig.26 (Adaptive Metasystem) which show graphs for order independent averages of u(n). The individual u values from which these graphs are derived appear in Table 14 (Free Learning) and Table 15 (Adaptive Metasystem).

2.5.17. Reduction in ℓ^1 Uncertainty Experienced by Subject

Since the adaptive metasystem is designed to present and work out relevant strategic alternatives, it is hypothesised that the percentage of probabilistic ℓ^1 statements will be smaller in the adaptive metasystem than it is in the free learning condition. This hypothesis is supported by the data in Table 16. The difference between the group percentages is significant at the 2% level, 0.002 > p.

2.5.18 Type of Mental Processing and the Form of Response

It is proposed that subjects can be divided into (1) Those who see things (the problem posing stimuli) as a whole and who construct a solution procedure as a whole and (2) Those who see things in parts and construct a sequential solution, bit by bit. Of these groups, the former (wholists) are probably apt to produce a "grouping" latency pattern and the latter a "string". There is some evidence that the "wholists" can be parodied as visual thinkers (all of the "grouping" subjects either mentioned or agreed to visual or spatial imagery), and that the "sequentialists" can be parodied as verbal or symbolic thinkers (all but one of them had a numbering scheme or a mnemonic).

There is insufficient data to reach a conclusion, but this matter deserves further investigation. It would be particularly interesting if, as may be hypothesised, the Type C strategy is fitted to the visual (wholist) subject and the Type A strategy to the verbal (sequentialist) subject.

TABLE 1.

	T(a)	T(b)	T(c)
	360	552	560
	424	576	424
	488	1136	400
	720	1176	488
	1064	1464	
	1448	1672	
Mean	750.7	1096	468
S.D.	425	457	72

Table 1. Trials to criterion, T, for subjects in the pilot experiment.

(a) = Unconstrained condition.
(b) = Constrained by procedure of 1.6.6.(II)
(c) = Simple adaptive system based on Type 0 strategy.

TABLE 2.

Complete K of R.		Partial K of R		Adaptive Metasystem.	
Subject Number	δt.	Subject Number.	δt.	Subject Number.	δt.
5	3.23	10	3.13	16	3.32
6	3.02	11	3.02	17	3.13
7	2.55	12	3.42	18	3.23
8	3.23	13	3.5	19	3.02
9	3.02	14	3.23	20	2.55
				21	3.13
Mean	3.01		3.26		3.06
S.D.	0.275		0.192		0.311

Table 2. Determined Values of δt.
Value of $\Delta t = \delta t + 2$ secs.

TABLE 3.

Bridging Subjects.			Complete K of R			Partial K of R			Adaptive Metasystem		
Subject Number	T_1	T_2	Sub. No.	T_1	T_2	Subj. No.	T_1	T_2	Subj. No.	T_1	T_2
1	43	311	5	329	513	10	147	380	16	297	220
2	189	754	6	236	558	11	121	300	17	184	286
3	112	855	7	77	249	12	302	748	18	184	276
4	297	915	8	181	623	13	183	610	19	207	180
			9	77	488	14	121	475	20	104	148
									21	133	209
Mean	160	709		180	486		175	543		185	220
S.D.	109	272		108	143		75	139		65	53

Table 3. Values of T for preliminary experiment, T_1 and for main experiment, T_2.

TABLE 4.

Complete K of R.			Partial K.of R.			Adaptive Metasystem		
Sub. No.	Blocks to Criterial level.	E	Sub. No.	Blocks to Criterial level	E	Sub. No.	Blocks to Criterial level.	E
5	37	14.35	10	28	14.4	16	19	11.6
6	37	14.56	11	32	15.6	17	22	13.0
7	18	13.83	12	31	24.3	18	23	12.0
8	36	17.31	13	30	20.3	19	16	11.2
9	37	13.19	14	30	15.8	20	13	11.4
						21	18	11.7
Mean		14.6			18.1			11.8
S.D.		1.56			4.12			0.58

Table 4. Mean Block Length, E, to Reach Criterial Performance in Main Experiment.

TABLE 5.

Complete K of R.		Partial K of R.		Adaptive Metasystem.	
Sub. No.	T* (%)	Sub. No.	T* (%)	Sub. No.	T* (%)
5	64.1	10	38.7	16	135.0
6	42.3	11	24.2	17	64.3
7	30.9	12	40.4	18	66.7
8	29.0	13	30.0	19	115.0
9	15.4	14	25.5	20	70.3
				21	64.6
Mean	30.3		31.8		86.0
S.D.	7.3		7.6		30.9

Table 5. Values of $T^* = T_1/T_2 \cdot 100\%$

TABLE 6.

Sub. No.	Including Responses to $x = X_{ABCD}$	Number of Response Sequences	Excluding Responses to $x = X_{ABCD}$	Number of Response Sequences
16	80	100	86	76
17	83.7	123	89.6	77
18	73.4	128	80.7	104
19	61.2	67	69.5	36
20	40.4	52	60.7	28
21	82.2	96	90.6	64
Mean	70.1	94.3	79.5	64.2
S.D.	12.0	30.3	12.1	14.3

Table 6. Consistency of Response Patterns in Adaptive Metasystem.

Table 7. The Number of Different Subproblem Classes Selected in Main Experiment, M, and Preliminary Exoeriment, P.

A			C			B		
	M	P		M	P		M	P
(7)	7	4	(6)	7	2	(5)	15	14
(8)	5	3	(12)	6	5	(10)	12	8
(11)	7	1	(9)	7	4	(13)	8	7
						(14)	11	7

Excluding A,B,C,D, Selections

A			C			B		
	M	P		M	P		M	P
(7)	4	4	(6)	3	1	(5)	11	10
(8)	4	3	(12)	3	5	(10)	8	7
(11)	7	1	(9)	5	4	(13)	8	6
						(14)	9	6

TABLE 8.

Type A Strategy		Type C Strategy		Type B Strategy	
Sub. No.	%	Sub. No.	%	Sub. No.	%
1	32	6	80	5	60
8	27	12	70	10	52
11	18	9	72	13	24
				14	57

Adaptive Metasystem.

16. 87% (chose Type C Strategy)
17. 76% (chose Type C Strategy)
18. 76% (chose Type C Strategy)
19. 16% (chose Type A Strategy)
20. 35% (chose Type A Strategy)
21. 90% (chose Type C Strategy.)

Note: Any occurrence of a pair as part of a string, i.e. strings of the form "Double Single" or "Single Double" is counted as a group. This elevates the percentage of groups differentially for Type A subjects and Type B subjects.

Table 8. Approximate Percentage of Grouped Responses excluding A,B,C,D, and ABCD (to nearest 1%.

TABLE 9.

Type A Strategy		Type C Strategy		Type B Strategy	
Sub. No.	%	Sub. No.	%	Sub. No.	%
7	93	6	100	5	55
8	100	12	89	10	91
11	86	9	100	13	88
				14	84

Table 9. Similarity Measure; After against Actual (to nearest 1%)

TABLE 10.

Type A Strategy		Type C Strategy		Type B Strategy	
Sub. No.	%	Sub. No.	%	Sub. No.	%
7	93	6	93	5	39
8	100	12	89	10	82
11	86	9	100	13	58
				14	53

Table 10. Similarity Measure: Before against Actual (to nearest 1%).

TABLE 11.

Type A Strategy		Type C Strategy		Type B Strategy.	
Sub. No.	%	Sub. No.	%	Sub. No.	%
7	100	6	93	5	48
8	100	12	100	10	82
11	86	9	100	13	58
				14	66

Table 11. Similarity Measure: Before against After. (to nearest 1%).

TABLE 12.

Type A Strategy		Type C Strategy		Type B Strategy	
Sub. No.	%	Sub. No.	%	Sub. No.	%
7	79	6	25*	5	95
8	100	12	75	10	65
11	28*	9	86	13	84
				14	77

*Subjects adopting all ABCD in Preliminary.

Adaptive Metasystem in Main Experiment.

16.	50%
17.	33%
18.	25%
19.	33%
20.	60%
21.	42%

Table 12. Similarity Measure: Preliminary against Main.

TABLE 13.

Partial K of R.		Complete K of R.		Adaptive Metasystem.	
Sub. No.	Mean of u(n)	Sub. No.	Mean of u(n)	Sub. No.	Mean of u(n)
5	0.84	10	0.7	16	0.44
6	0.77	11	0.95	17	0.57
7	0.73	12	1.9	18	0.49
8	1.27	13	1.64	19	0.52
9	0.71	14	1.0	20	0.50
				21	0.44
Mean	0.86		1.24		0.49
S.D.	0.23		0.50		0.08

Table 13. Mean Values u(n).

TABLE 14: Complete K.R.

Subject No.5			Subject No.6			Subject No.7			Subject No.8			Subject No.9		
n	i	$u_i(n)$	n	i	$u_i(n)$	n	i	$u_i(n)$	n	i	$u_i(n)$	n	i	$u_i(n)$
1	A	0.4	1	A	0.4	1	A	1.5	1	AB	2.1	1	A	0.1
2	B	0	2	B	0.4	2	B	0.3	2	A	0.1	2	AB	0.4
3	AB	1.5	3	B	0.1	3	C	0.5	3	AB	0.4	3	C	0.4
4	C	0;1	4	AB	2.0	4	ABC	2.8	4	ABC	1.9	4	C	0.1
5	D	0.4	5	AB	0.8	5	AB	0.1	5	ABC	1.5	5	AB	0.1
6	AB	1.0	6	C	0.1	6	ABC	0.9	6	ABC	1.4	6	ABC	1.3
7	AC	1.6	7	D	0.1	7	ABC	0.6	7	ABC	1.0	7	AB	1.0
8	AC	0.5	8	A	0.3	8	ABC	0.1	8	ABC	1.9	8	CD	0.3
9	BC	1.1	9	B	0	9	BCD	0.8	9	ABC	0.5	9	ABC	1.6
10	AD	0.9	10	ABCD	6.4	10	BCD	0.6	10	ABC	0.4	10	AB	0.1
11	BD	0.3	11	A	0.1	11	BCD	0.5	11	ABCD	1.5	11	ABC	0.6
12	CD	0	12	B	0	12	ABCD	1.1	12	ABCD	0.9	12	ABC	1.3
13	ACD	1.1	13	C	0.1	13	ABCD	1.9	13	ABCD	2.5	13	ABC	0.6
14	ACD	0.9	14	D	0	14	ABCD	0.6	14	CD	0.4	14	ABC	0.6
15	ABC	1.9	15	ABCD	4.9	15	ABCD	0.3	15	CD	0.4	15	BCD	1.6
16	ABC	1.5	16	ABCD	1.1	16	ABCD	0.5	16	ABCD	1.9	16	AB	0.4
17	ABC	2.0	17	ABCD	4.0	17	ABCD	0	17	ABCD	1.6	17	ABC	0.8
18	ABD	0.4	18	AB	0.4	18	ABCD	0.1	18	ABCD	1.3	18	ABC	0.1
19	BCD	1.5	19	AB	0.3				19	ABCD	1.8	19	ABC	0.9
20	BCD	0.5	20	AB	0.3				20	ABCD	0.8	20	ABC	0.5
21	ABCD	1.1	21	AB	0.1				21	ABCD	1.6	21	BCD	0.9
22	ABCD	0.8	22	AB	0.1				22	ABCD	1.6	22	BCD	0.6
23	ABCD	0.9	23	AB	0				23	ABCD	1.3	23	BCD	1.0
24	ABCD	0.9	24	CD	0.4				24	ABCD	2.1	24	ABC	0.4
25	ABCD	1.0	25	CD	0				25	ABCD	1.5	25	BCD	1.1
26	ABCD	1.0	26	CD	0				26	ABCD	1.1	26	BCD	0.6
27	ABCD	0.5	27	ABCD	1.5				27	ABCD	2.3	27	BCD	0.6
28	ABCD	1.5	28	ABCD	1.0				28	ABCD	0.9	28	ABC	0.5
29	ABCD	0.9	29	ABCD	2.6				29	ABCD	0.6	29	BCD	0.4
30	ABCD	1.1.	30	ABCD	0.8				30	ABCD	1.0	30	ABC	0
31	ABCD	0.6	31	ABCD	0.4				31	ABCD	0.8	31	ABCD	0.6
32	ABCD	0.8	32	ABCD	0.1				32	ABCD	1.3	32	ABCD	0.6
33	ABCD	0.6	33	ABCD	0.5				33	ABCD	0.8	33	ABCD	0.4
34	ABCD	0.1	34	ABCD	0.2				34	ABCD	0.5	34	ABCD	1.9
35	ABCD	0.4	35	ABCD	0.3				35	ABCD	0.4	35	ABCD	0.9
			36	ABCD	0				36	ABCD	0.3	36	ABCD	0.4
			37	ABCD	0							37	ABCD	0.4

Table 14. Complete $u_i(n)$ Values for Free Learning Subjects.

TABLE 14 (contd.): Partial K.R.

Subject No.10			Subject No.11			Subject No.12			Subject No.13			Subject No.14		
n	i	$u_i(n)$	n	i	$u_i(n)$	n	i	$u_i(n)$	n	i	$u_i(n)$	n	i	$u_i(n)$
1	A	1.1	1	A	0.4	1	A	0.9	1	AB	3.5	1	A	0.4
2	B	0.8	2	AB	2.3	2	B	0.6	2	CD	2.9	2	AB	1.0
3	B	0.1	3	A	0	3	AB	4.1	3	BC	3.4	3	AB	0.3
4	C	0.4	4	AB	0.5	4	AB	2.1	4	BC	6.4	4	CD	2.5
5	D	0.1	5	ABC	3.8	5	C	0.6	5	BC	0.8	5	C	0.4
6	B	0	6	CD	0.9	6	CD	1.6	6	BC	1.0	6	CD	0.4
7	AB	0.8	7	BC	1.3	7	AB	0.1	7	CD	0.8	7	BC	1.4
8	AB	0	8	ABCD	1.8	8	ABCD	10.9	8	BCD	1.8	8	AD	0.1
9	C	0.3	9	ABCD	1.0	9	AB	0.1	9	BCD	1.3	9	ABCD	8.1
10	D	0.3	10	ABCD	1.4	10	AB	0.1	10	BCD	0.9	10	AC	0.5
11	ABCD	6.0	11	ABCD	1.5	11	CD	0.5	11	ABC	1.9	11	BD	0.5
12	C	0.3	12	ABCD	1.3	12	CD	2.1	12	ABC	1.3	12	BCD	1.1
13	D	0	13	ABCD	1.4	13	ABCD	11.1	13	ABC	1.6	13	BCD	0.8
14	AC	0.4	14	ABCD	1.4	14	AB	1.0	14	ABC	1.5	14	BCD	0.8
15	AC	0	15	ABC	0.5	15	CD	0.1	15	ABC	0.7	15	CD	0.1
16	AD	0.1	16	BCD	3.4	16	ABCD	5.4	16	ABC	0.7	16	ABC	0.5
17	BC	0.3	17	ABC	0.9	17	ABCD	2.4	16	ABD	0.6	17	BCD	0.5
18	BD	0.8	18	ABC	0.6	18	ABCD	1.0	18	ABD	0.7	18	BCD	0.8
19	CD	0	19	ABC	0.1	19	ABCD	1.1	19	ACD	0.8	19	BC	0.1
20	ABC	0.6	20	ABC	0.1	20	ABCD	1.4	20	ACD	2.0	20	BCD	0.3
21	ABC	1.4	21	ABC	1.3	21	ABCD	1.0	21	ACD	1.9	21	ABCD	2.5
22	ABC	0.9	22	ABC	0.4	22	ABCD	2.0	22	ACD	0.3	22	ABCD	0.6
23	ABC	0.5	23	ABC	0.1	23	ABCD	2.3	23	ABCD	2.1	23	ABCD	0.6
24	ABC	0.8	24	ABCD	0.9	24	ABCD	3.4	24	ABCD	1.7	24	ABCD	1.5
25	ABC	0.5	25	ABCD	0.9	25	ABCD	1.9	25	ABCD	3.3	25	ABCD	1.5
26	ABCD	2.4	26	ABCD	0.4	26	ABCD	1.3	26	ABCD	1.8	26	ABCD	1.1
27	ABCD	0.3	27	ABCD	0.6	27	ABCD	0.3	27	ABCD	0.8	27	ABCD	0.6
28	ABCD	0.4	28	ABCD	1.1	28	ABCD	0.4	28	ABCD	1.7	28	ABCD	0.5
			29	ABCD	0	29	ABCD	1.4	29	ABCD	0.6	29	ABCD	0.1
			30	ABCD	0.3	30	ABCD	0.3	30	ABCD	0.4	30	ABCD	0.4
			31	ABCD	0.1	31	ABCD	0.4						
			32	ABCD	0									

Table 14. Complete $u_i(n)$ Values for Free Learning Subjects. (contd).

TABLE 15. Adaptive Metasystem.

Subject No.16			Subject No.17			Subject No.18			Subject No.19			Subject No.20			Subject No.21		
n	i	$u_i(n)$	n	i	$u_i(n)$	n	i	$u_i(n)$	n	i	$u_i(n)$	n	i	$u_i(n)$	n	i	$u_i(n)$
1	A	0.1	1	A	0	1	A	0	1	A	0	1	A	0.1	1	A	0.1
2	B	0.3	2	B	0	2	AB	0.8	2	B	0.1	2	AB	0.8	2	B	0.1
3	AB	0.5	3	AB	1.5	3	AB	0.3	3	AB	0.4	3	AB	0.5	3	AB	1.1
4	AB	0.3	4	AB	0.1	4	C	0	4	C	0	4	A	0.1	4	B	0
5	AB	0.1	5	C	0	5	CD	2.4	5	ABC	1.9	5	B	0.3	5	AB	0.1
6	C	0.1	6	D	0	6	D	0.5	6	ABC	0.5	6	AB	0	6	C	0.1
7	D	0.1	7	CD	0.9	7	C	0	7	ABC	0.3	7	ABC	0.1	7	CD	0.1
8	CD	0.4	8	AB	0	8	CD	0.1	8	D	0	8	ABCD	2.1	8	CD	0.4
9	CD	0.5	9	CD	0	9	AB	0.6	9	ABCD	1.3	9	ABCD	1.5	9	AB	0.1
10	CD	0.3	10	ABCD	3.8	10	CD	1.0	10	D	0	10	D	0.3	10	CD	0
11	AB	0	11	AB	0.5	11	AB	0.5	11	ABC	0.4	11	ABC	0.1	11	CD	0.1
12	ABCD	4.1	12	CD	0.1	12	CD	0.6	12	ABC	0.3	12	ABCD	0.4	12	ABCD	2.8
13	CD	0.1	13	ABCD	2.6	13	AB	0.3	13	ABCD	0.6	13	ABCD	0.4	13	AB	0
14	AB	0	14	CD	0.3	14	CD	0	14	ABCD	0.4				14	CD	0.1
15	CD	0.1	15	AB	0	15	AB	0.1	15	ABCD	0.4				15	ABCD	1.4
16	AB	0.3	16	ABCD	1.9	16	AB	0	16	ABCD	0.1				16	ABCD	1.1
17	ABCD	0.8	17	ABCD	1.4	17	CD	0							17	ABCD	0.1
18	ABCD	0	18	CD	0	18	ABCD	2.6							18	ABCD	0.3
19	ABCD	0.3	19	AB	0	19	AB	0.3									
			20	ABCD	0.4	20	CD	0									
			21	ABCD	0.4	21	ABCD	0.6									
			22	ABCD	0.1	22	ABCD	0.4									
						23	ABCD	0.3									

Table 15. Complete $u_i(n)$ Values for Adaptive Metasystem Subjects.

TABLE 16.

Sub. No.	G. %	Sub. No.	G. %	Sub. No.	G. %
5	23	10	4	16	0
6	11	11	22	17	5
7	44	12	3	18	0
8	8	13	7	19	0
9	32	14	13	20	8
				21	0
Mean:	23.6		9.8		2.2
S.D.	14.9		7.9		8.1

Table 16. The Percentage, G, of ℓ^1 Probabilistic Statements amongst ℓ^1 Statements (to nearest 1%).

FIGURE 18.

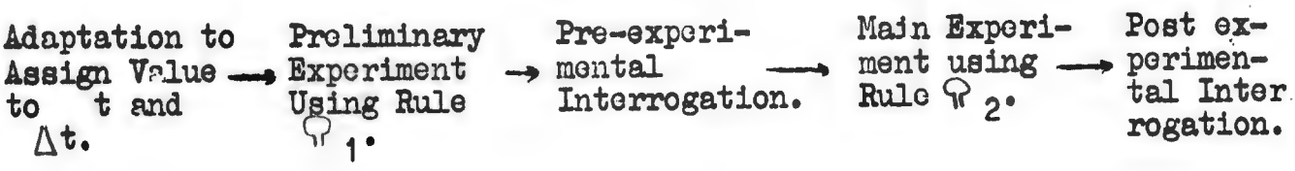


Figure 18. Procedure for the Principal Experiment.

Figure 19

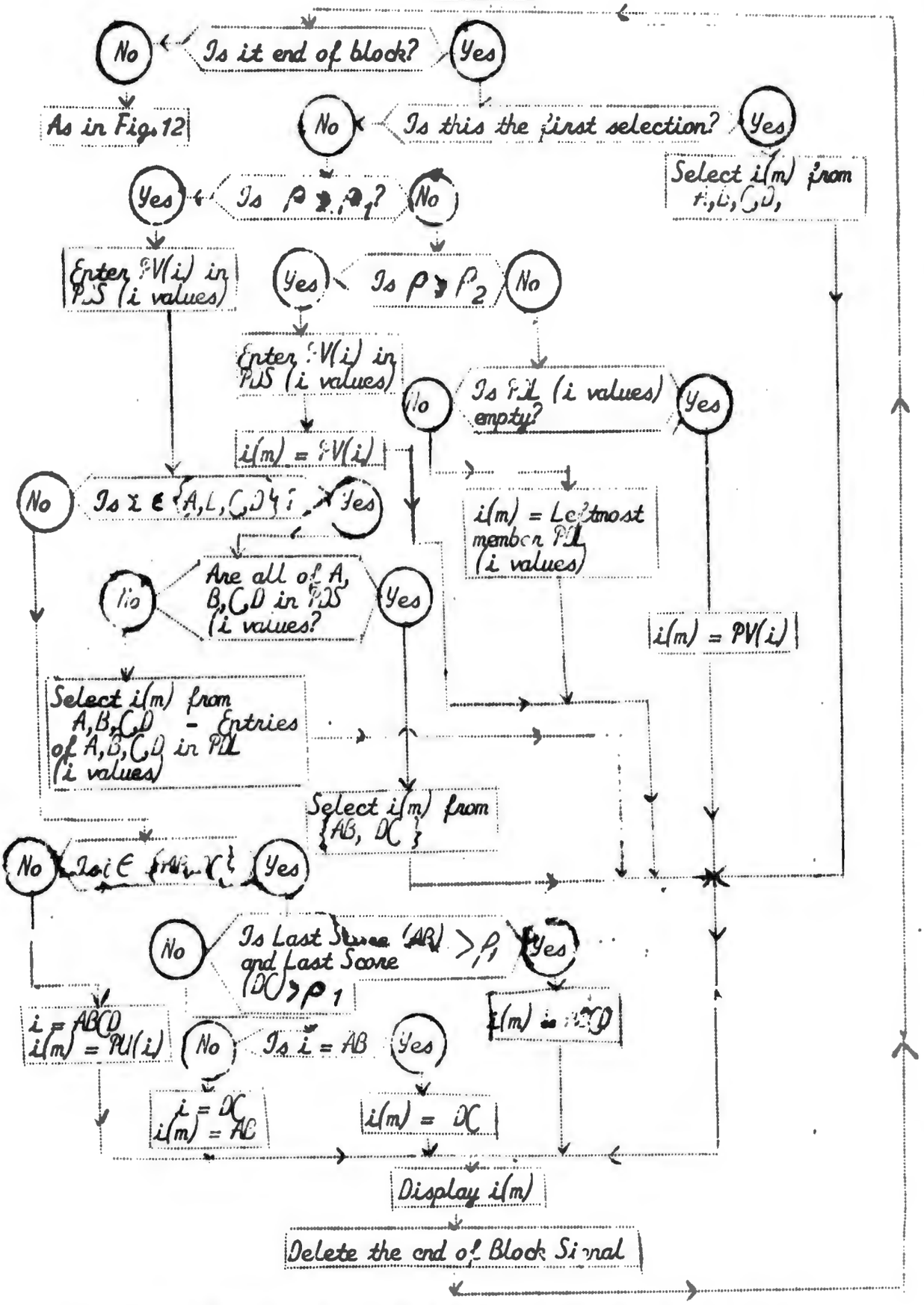


Figure 19. Flow Chart for a simple adaptive teaching machine using Strategy C. Only the end of block processing is shown since the rest of the action of the automaton is identical with Figure 12. PV(i) = Present Value of i. PDL (i values) is a push down list of previous i values. When a symbol is read out of this list it is deleted by the system.

FIGURE 20.

a,b,.....stimulus lamps. 1,2,.....Response keys.

Ω_1 = 1,a; 1,e; 3,c; 4,g; 5,b; 6,f; 7,d; 8,h.

Figure 20. Rule Used in Preliminary Experiment.

FIGURE 21.

a,b,.....stimulus lamps. 1,2,.....Response keys.

Ω_2 = 1,h; 1,c; 3,f; 4,a; 5,e; 6,b; 7,g; 8,d.

Figure 21. Rule used in Main Experiment.

Fig.22(i). Common Stages for Each Strategy.

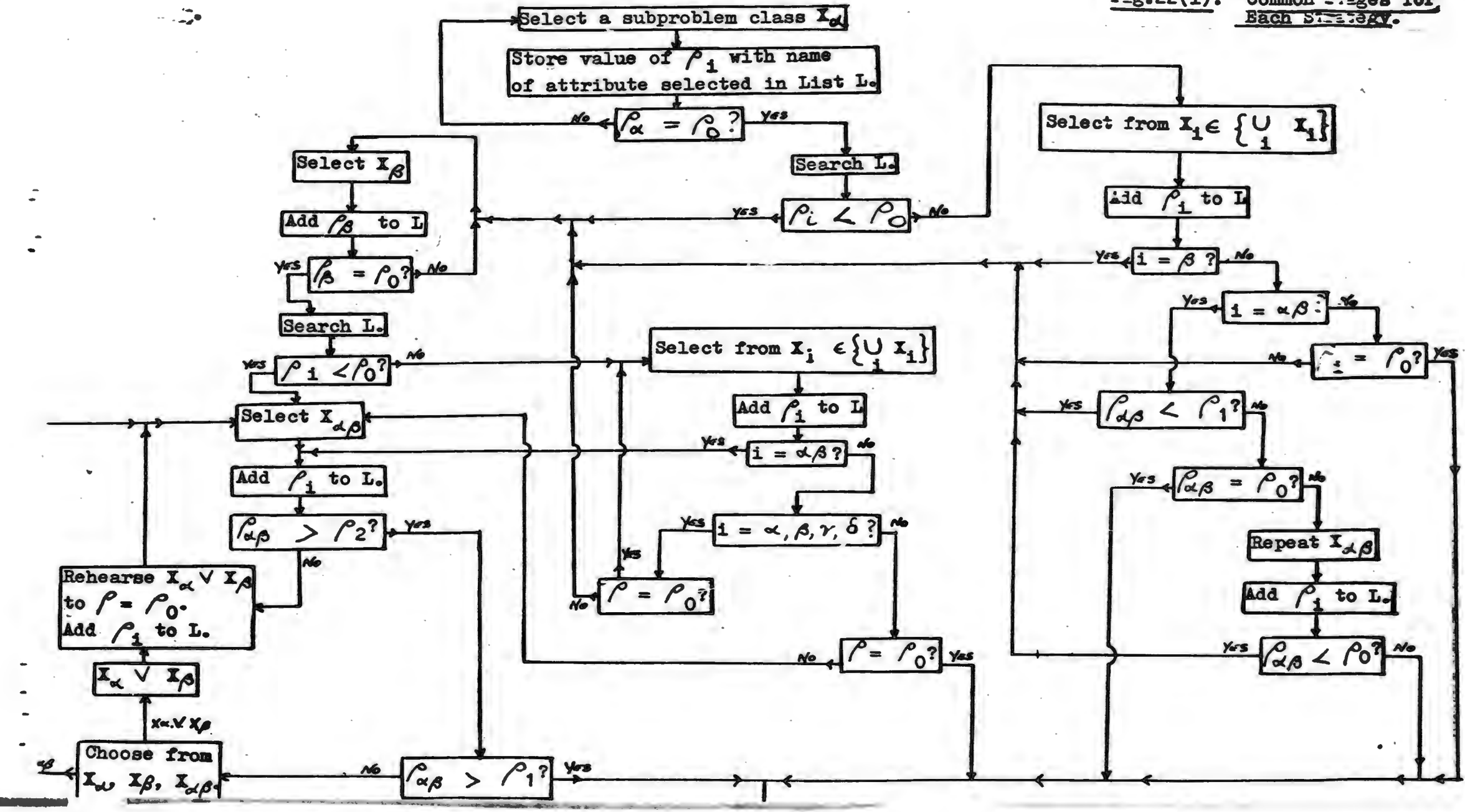


Figure 22(i) (contd.)

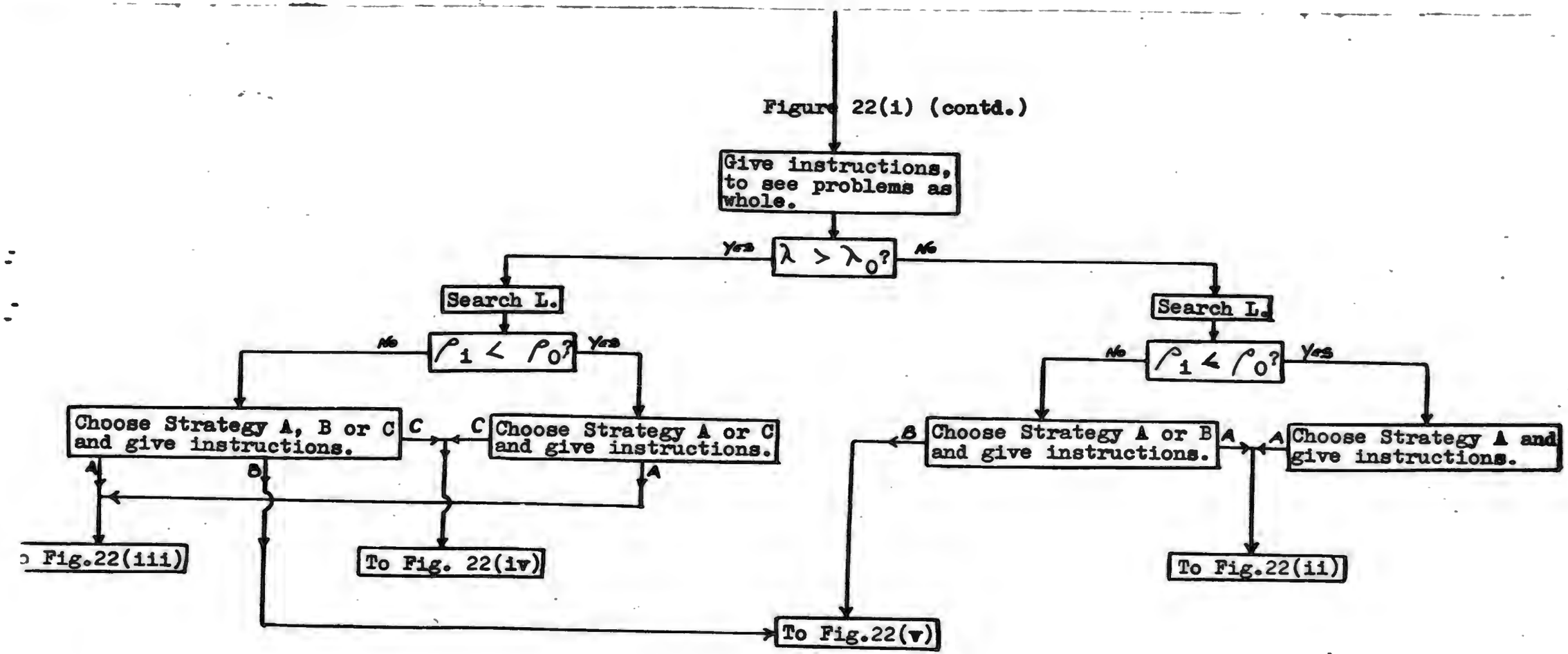


Figure 22(i)

Figure 22(ii)

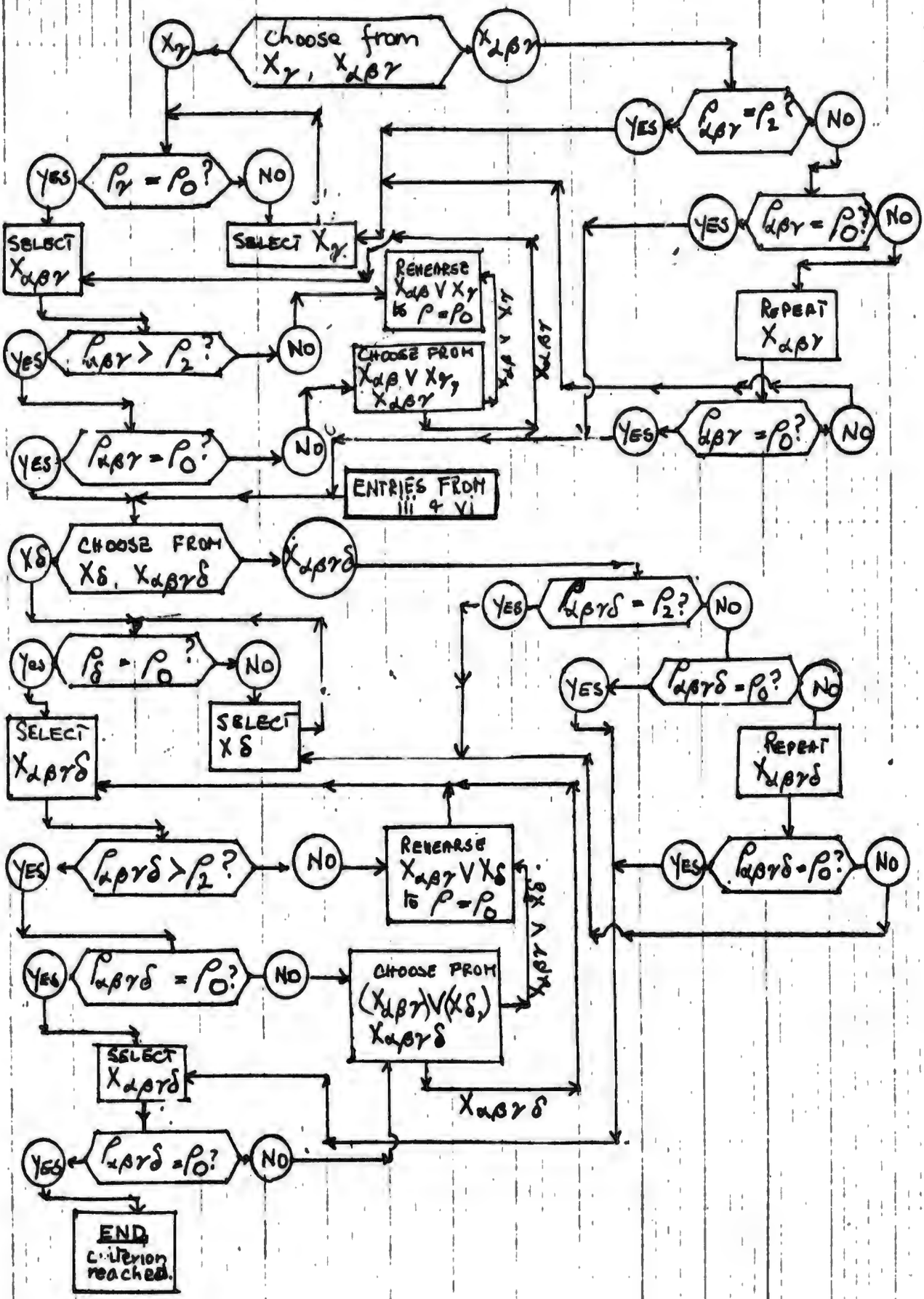


Figure 22(ii). Strategy A (for $\lambda < \lambda_0$).

Figure 22(iii)

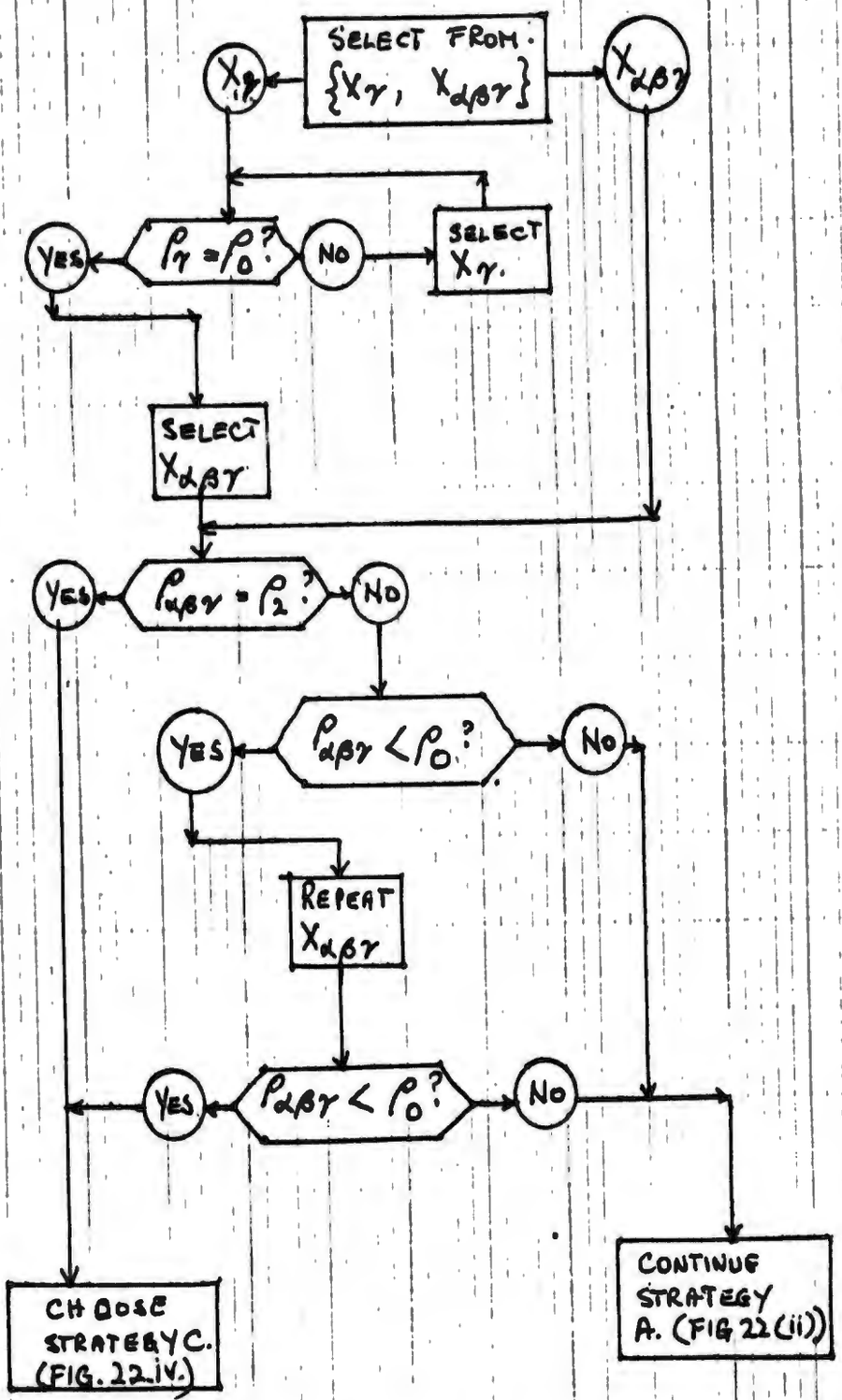


Figure 22(iii). Strategy A (for $\lambda > \lambda_0$)

Figure 22(iv)

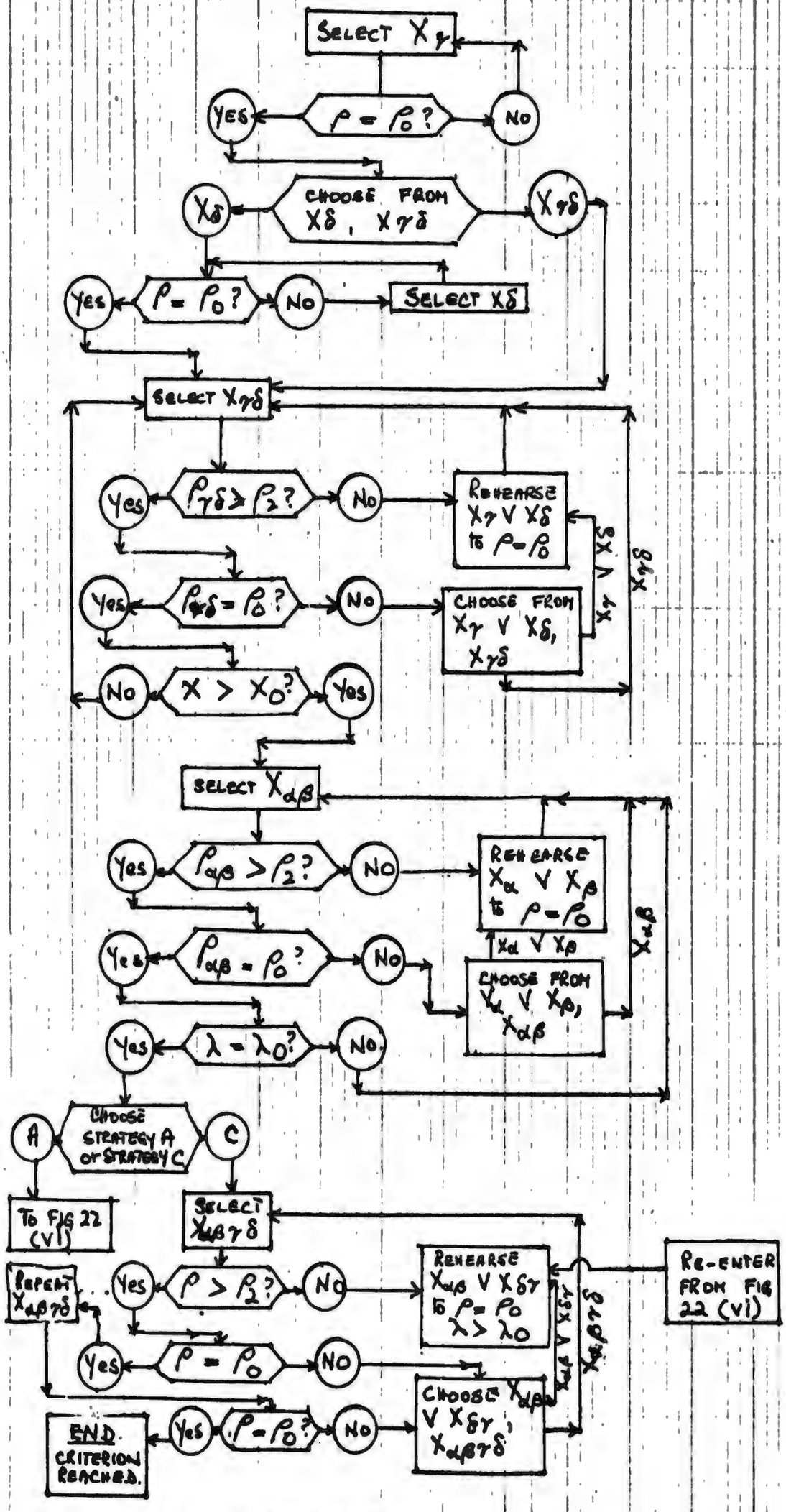


Figure 22(iv). Strategy C.

Figure 22(v)

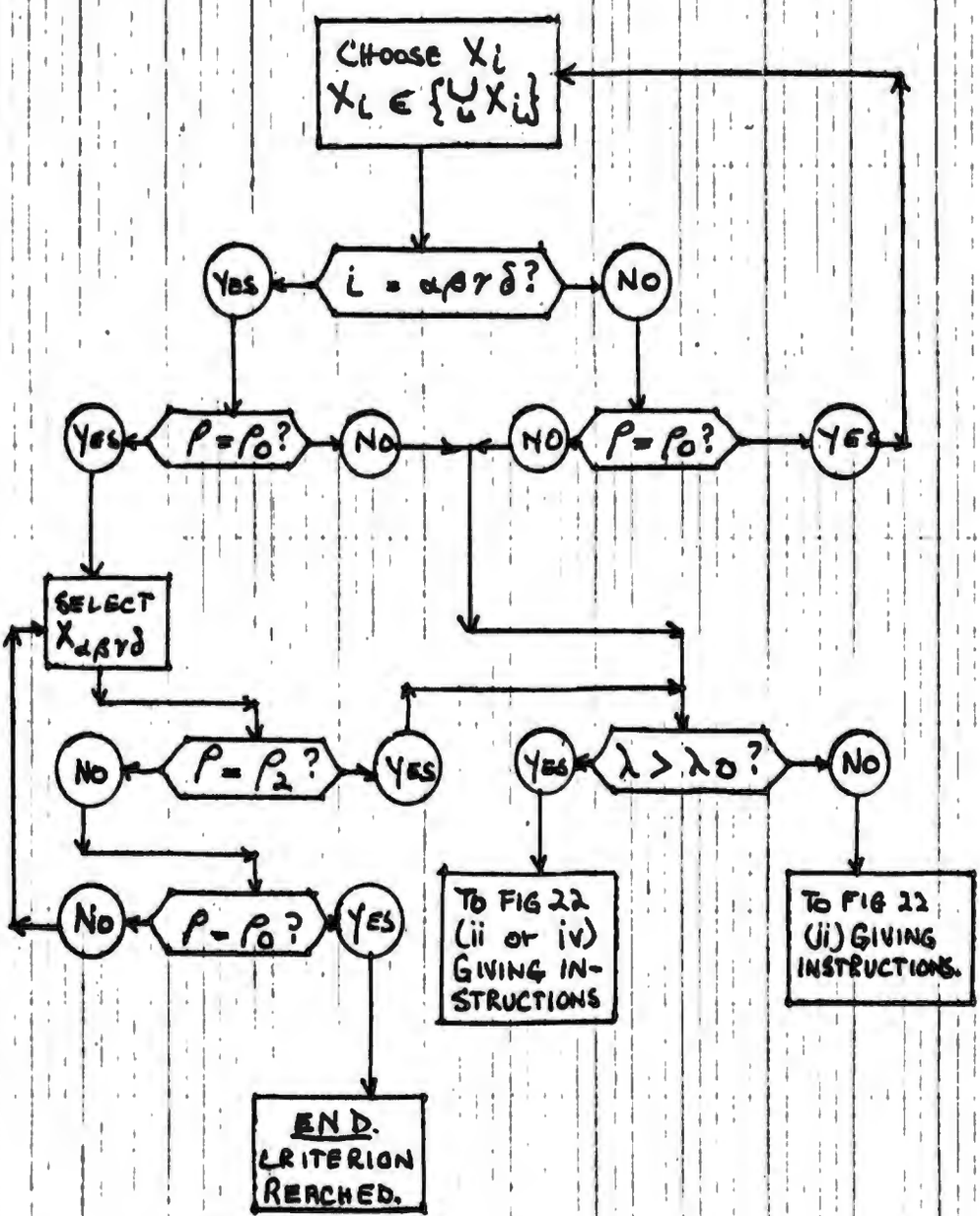


Figure 22(v). Strategy B.

Figure 22(vi)

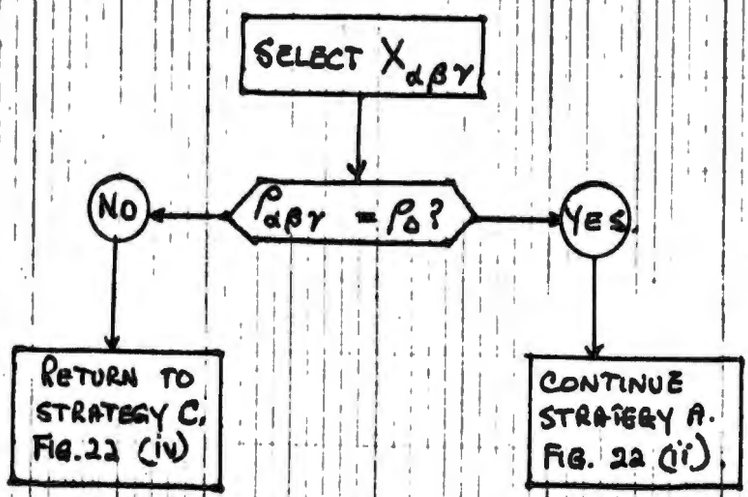


Figure 22(vi) Modification of Strategy to Strategy A.

FIG. 23 REPETITIONS (With ambiguous subjects.)

Histogram Showing Frequency of Selections ("ambiguous" subjects included)

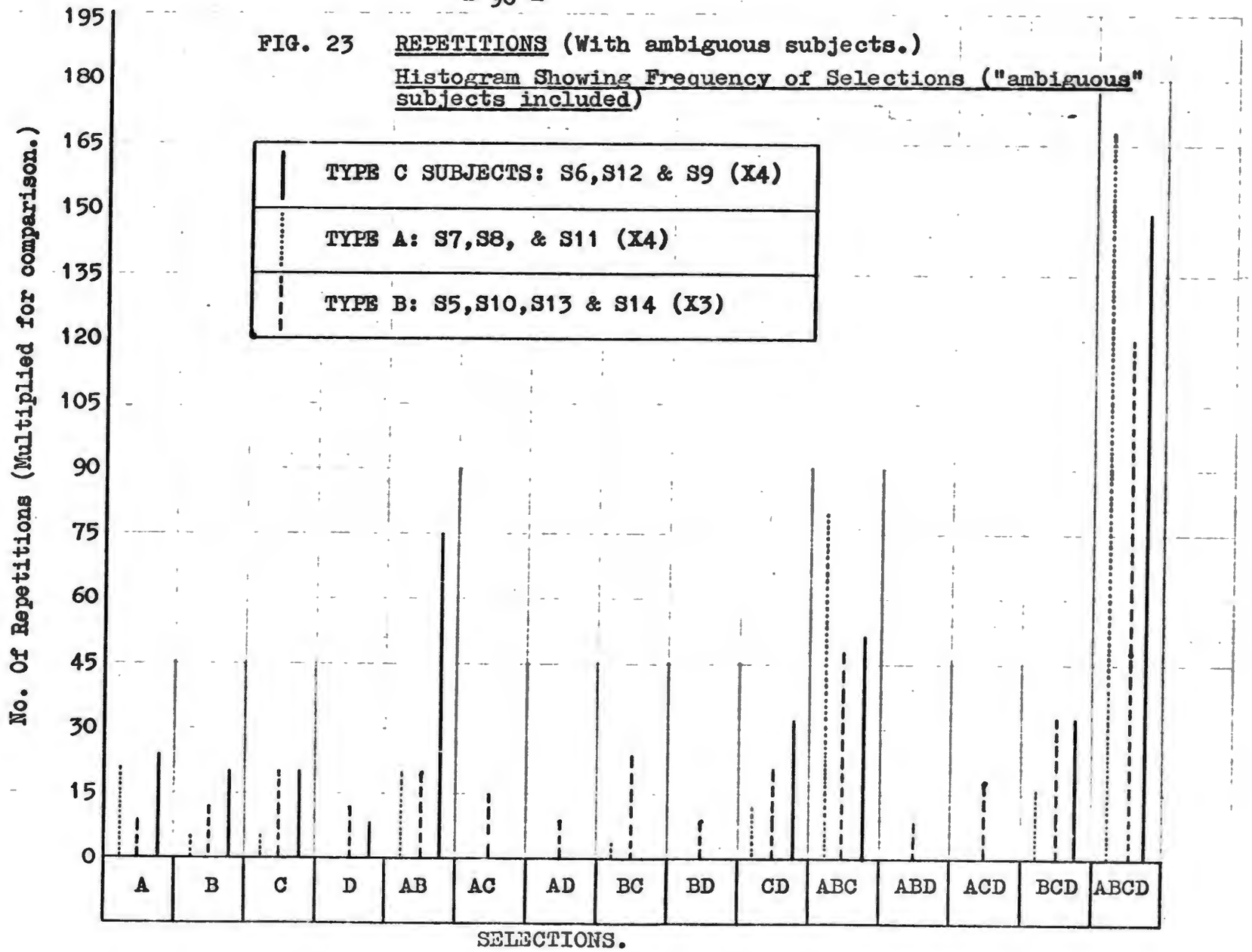
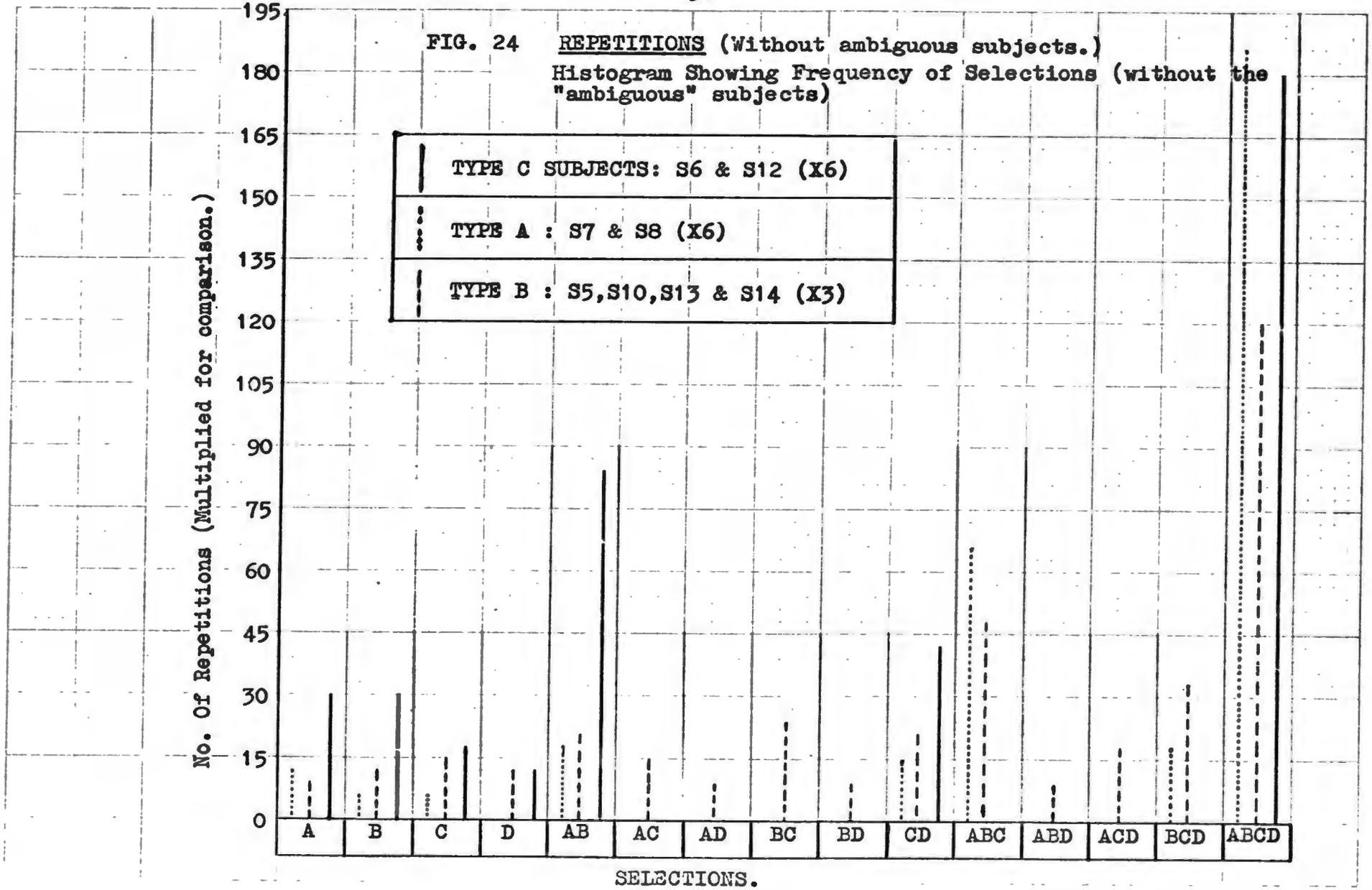


FIG. 24 REPETITIONS (Without ambiguous subjects.)
Histogram Showing Frequency of Selections (without the "ambiguous" subjects)



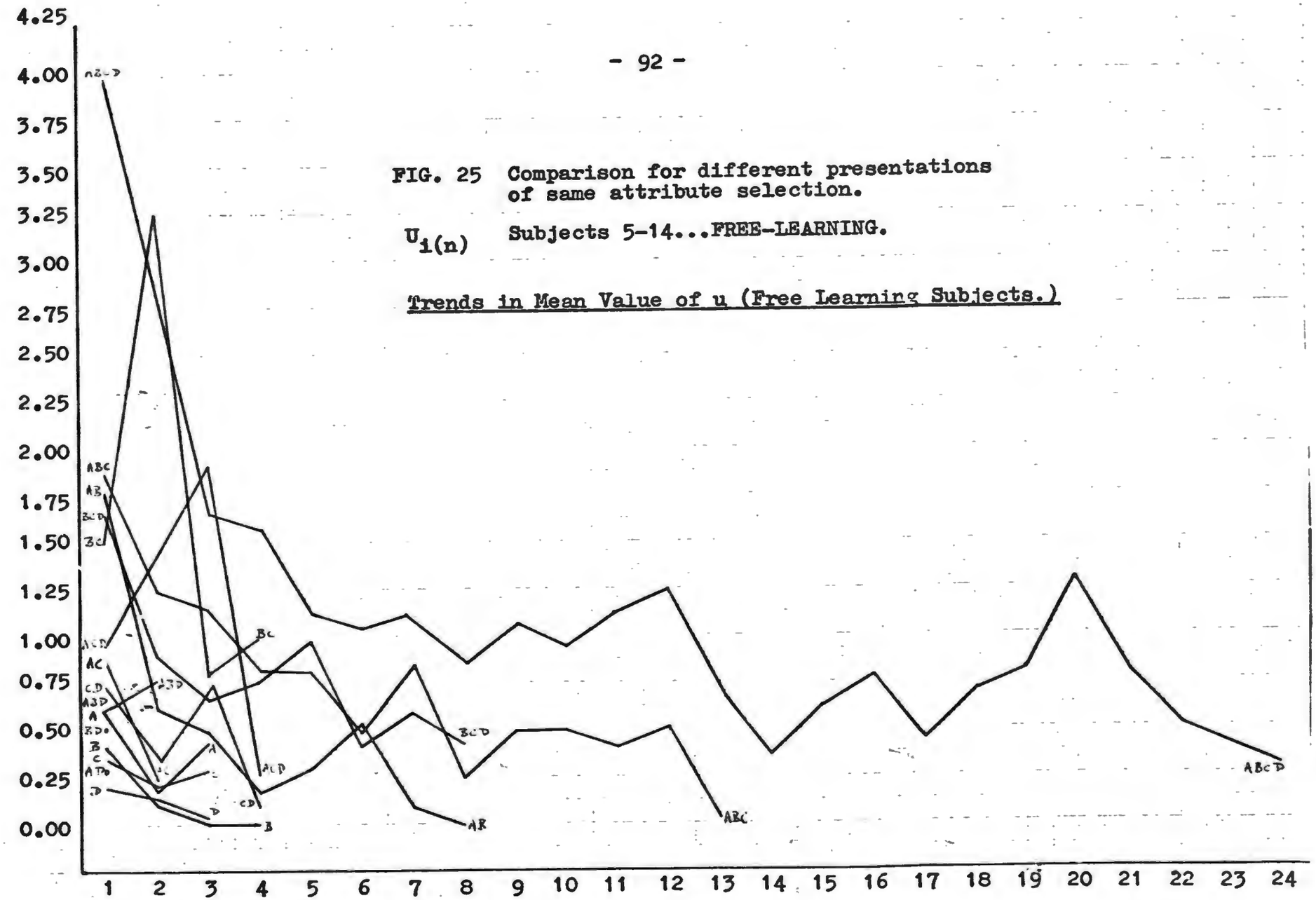


FIG. 25 Comparison for different presentations of same attribute selection.

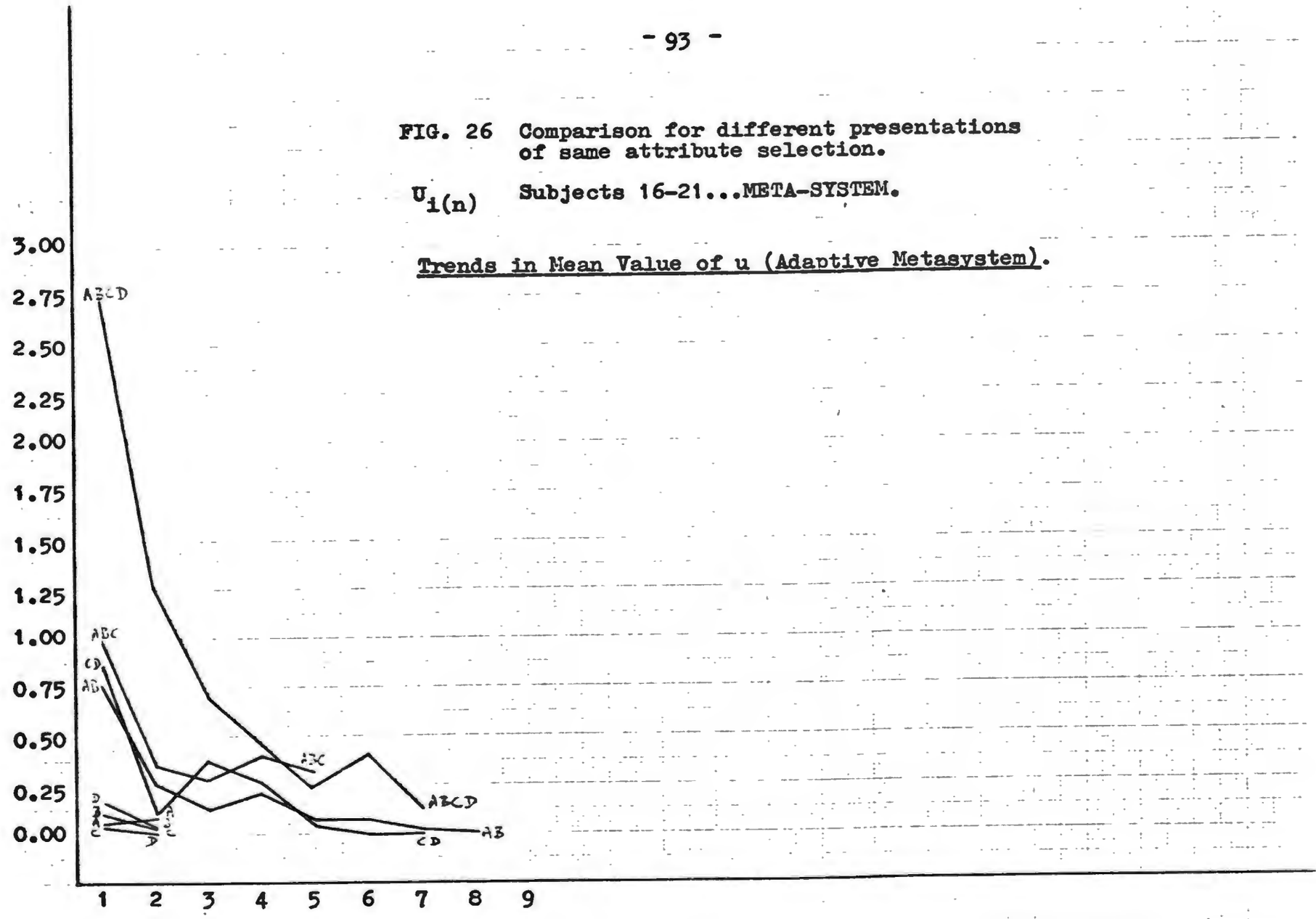
$U_{i(n)}$ Subjects 5-14...FREE-LEARNING.

Trends in Mean Value of u (Free Learning Subjects.)

FIG. 26 Comparison for different presentations of same attribute selection.

$U_{i(n)}$ Subjects 16-21...META-SYSTEM.

Trends in Mean Value of u (Adaptive Metasystem).



SECTION : 3

LEARNING AS THE DEVELOPMENT OF A SELF-ORGANISING SYSTEM

3. Learning as the Development of a Self-Organising System

At a macroscopic level, the operation of a teaching system, or, for that matter, a free learning process, can be described in terms of Von Foerster's theory of self-organisation; a description of a simple adaptive system, couched in these terms, is given in Ref. (9). Due to the difficulties involved in setting up stochastic equations that respect the complexities of free learning or learning in a metasystem, it is better to make predictions at a macroscopic level, using a computer programmed learning model as the predictive instrument. Such a model is presented in Section 5. All the same, the macroscopic model can be usefully checked against reality since the variable $u(n) = u(t)$ is a good estimate of one of the main variables (H, as below) in a self-organising system model. Thus, an overall "self-organisational" description will be given chiefly for comparative purposes (no attempt is made to set up stochastic system equations, though this might be done in principle).

According to Von Foerster's theory, Ref.10, redundancy, R, is a suitable measure of organisation -

$$R = 1 - H/H_{max}$$

Here H is an information measure on a System, S, and H_{max} is its maximum value.

$$H = - \Sigma_r p_r \text{ Log } p_r$$

where p_r is the probability of occurence of the r^{th} state.

In general, H_{max} depends upon the structural description of the system (in the simplest case, if there are M possible states, $H_{max} = -$ Log M); H_{max} will change in value insofar as this structure changes. On the other hand, H is a behavioural measure; it will decrease if the system adapts and increase if its behaviour becomes irregular. Von Foerster proposes -

$$dR/dt > 0 \text{ ..... Eq.(1)}$$

as the most suitable criterion for self-organisation. From the definition of a redundancy, Eq.(1) may be written as -

$$\left(\frac{-H \frac{dH_{max}}{dt} - H_{max} \frac{dH}{dt}}{H_{max}^2} \right) > 0$$

Disregarding the trivial case of $H_{max} = 0$, and dividing through by $-H.H_{max}$, we obtain, with Von Foerster, the basic condition for self-organisation -

$$\frac{1}{H_{max}} \frac{dH_{max}}{dt} > \frac{1}{H} \frac{dH}{dt} \text{ ..... Eq.(2)}$$

or, equivalently -

$$H \frac{dH_{max}}{dt} > H_{max} \frac{dH}{dt}$$

3.1. One Interpretation

We shall now conceive the subject in contact with the experimental environment (either of the free learning situation or the

or the adaptive metasystem) as a system, S. From the discussion in Section 1 and Section 2, it will be clear that S is made up from subsystems, S_i, such that the state set of S_i is X_i, Y_i (where X_i has the meaning established before and Y_i is the set of $\mathcal{L}^0$ legal responses to $x \varepsilon X_i$).

The subject sets out to investigate and learn about the problem environment equipped with a strategy, which we shall assume remains unchanged throughout the learning process*. At the outset, the subject has available certain a priori values of $H_{max}(S_i)$, i.e. the values of H_{max}, which would apply if the subproblem class were selected. In general, these have the form $-\text{Log}\binom{8}{m}$ per stimulus (there being m attributes in a specification of the relevant subproblems). In fact, we can have little faith in these values because of the internal and temporal constraints entailed in the task, but it is safe to say -

$$H^*_{max}(X_iY_i) = H^*_{max}(X_jY_j)$$

if X_i and X_j denote problems at the same level in the hierarchy of problem classes and that -

$$H^*_{max}(X_iY_i) > H^*_{max}(X_j,Y_j) \quad \ldots\ldots\ \text{Eq.(3)}$$

if the class designated by X_i is at a higher level in the hierarchy than X_j. In any case, the values of $H_{max}(S_i)$ represents the subject's maximum uncertainty about what he is allowed to do (by way of problem solving, given the vocabulary and the rules of $\mathcal{L}^0$, when S_i is selected). Clearly, when he begins to learn about the problems belonging to X_i, he will experience some actual uncertainty, H(i), and it is proposed (for the purpose of an idealised model) that $H_{max}(X_iY_i)$ has the form -

$$H_{max}(t) = H_{max}(X_i,Y_i,t) = H^*_{max}(X_iY_i)$$

if X_iY_i has not yet been selected

$$= \text{LAST VALUE } H(i) + \lambda(t-t(i))$$

if X_i was last selected at t(i) ..... Eq.(4)

FOOTNOTE: *Type B strategies are, thus, excluded from this discussion.

(the term $\lambda(t-t_1)$ pays an idealised deference to forgetting; to approximate reality a further selection dependent term should be added to account for the effect of interference).

The subject's actual uncertainty, H, is his uncertainty about how to solve problems in an $\mathcal{L}^0$ legal and Ω satisfying fashion. Its value might be obtained by averaging over conditional response probabilities or their frequency estimates[α]. Fortunately, however, a direct estimate of this quantity is provided by u(n) = u(t) of Section 1.7.

It is argued that human beings are constructed to act as self-organising systems with respect to their environment. This they do (1) by directing their attention to various S_i(or X_iY_i) at discrete instants $t = M\Delta t$[β] and, thus, controlling H_{max} and (2) Once a field of attention is selected, by engaging in goal directed adaptation, which reduces H. When a human being is left on his own, in the free learning condition of the present experiment, self-organisation is liable to break down and learning is impaired. The automaton in an adaptive metasystem is intended to prevent this sort of breakdown by mediating in the attention directing process.

FOOTNOTE:

[α] Some caution is called for in the interpretation of H. Obviously, from the definition of H given above

$$dH/dt = \sum_r dp/dt - \sum_r dp_r/dt \text{ Log } p_r$$

Since $\sum_r p_r = 1$ (as the p_r probabilities), $\sum_r dp_r/dt = 0$

thus, $dH/dt = -\sum dp_r/dt \text{ Log } p_r$

To obtain an estimate of H, it is possible to substitute the p_r by the conditional response frequencies p(y|x) pertinent to a selection X (where Y_1 is the set of legal responses to $x \in X$)

$$dH/dt = -\sum_{x,y \in X,Y_1} dp(y|x)/dt \text{ Log } p(y|x)$$

So that any behavioural convergence whatever counts as an adaptation. In learning, on the other hand, the adaptive process is goal directed, i.e. only one of the terms in this equation really has a consistently positive derivative, namely the term -

$$p(y|x) = p(y = \Omega(x)|x).$$

[β] assuming the values n(M) of the Block Ends, M.

3.2. An Arbitrary Idealisation

To see what may be involved in greater detail, suppose that X_1Y_1 is selected and consider an idealised model in which dH/dt is a function of H. The simplest form of this model postulates an upper limit of uncertainty, H_1, above which learning does not take place (overload occurs or the problems are too difficult to solve) and a lower limit, H_2, below which there is insufficient to learn about. Thus -

$$0 > dH/dt \text{ if } H_1 > H > H_2 > 0 \quad \ldots\ldots\ldots \text{Eq.(5)}$$

and if H_0 represents the initial uncertainty (prevailing at the instant when a subproblem class is selected) it is crucial that classes are selected such that -

$$H_1 > H_0 > H_2 > 0$$

Further, given this restriction, it is desirable for H_0 to be as near to H_1 as possible.

Again, in an ideal case, selection of a next subproblem class for attention might be determined by the flow chart of Fig.27, which describes a procedure carried out at the end of each block. (i.e. at instants $n\Delta t$ such that $n = r(M)$ for the M^{th} selection)

The learning process, thus, consists of a performance phase, in which problems are being solved and a selection phase, occupying a finite interval, τ say, in which the subject thinks out what sub-problem class to select and selects it. For the performance phase, $dH_{max}/dt = 0$ so that the left hand side of Eq.(2) is zero valued and the system, S, is a self-organising system if Eq.(5) is satisfied (so that $0 > dH/dt$). Conversely, in the selection phase, $dH/dt = 0$ and the system is a self-organising system insofar as the value of dH_{max}/H is increased by the selection.

An alternative and more sophisticated interpretation of the system is also possible.

Let $\overline{H}_{max}$ = Average value of $H_{max}(S_i)$

Let $\overline{\overline{H}}_{max}$ = smoothed value of H

Call the resulting system, $\overline{S}$, the smoothed system of S and observe that $\overline{S}$ is a self-organising system if it satisfies Eq.(2) with $\overline{H}_{max}$ substituted for H_{max} and $\overline{H}$ for H.

3.3. Relevant Hypotheses

In either case (since the excursion of H_{max} is limited by the task) self-organisation will involve a negative trend in $\partial H/\partial t$. This is generally observable (as estimated by the trend in u), but there are certain exceptions in the free learning experiments. Hence, positive values of $\partial H/\partial t$ occur (especially in connection with the subproblem classes, ABCD, ABC...) and give rise to prolonged peaks and periods of uncertainty, i.e. the mean value of u is elevated when the deviant behaviour is manifest. Insofar as the adaptive metasystem is designated to improve the choice process of the flow chart, and, consequently, to secure "better" self-organisation, it is predicted (1) that deviant behaviour will be absent from the metasystem and (2) that the mean value of u will be consistently lower than it is in a free learning experiment (2.5.14., 2.5.15., and 2.5.16).

So far, we have considered self-organisation with respect to level $\mathcal{L}^0$ only. However, a comparable process takes place at level $\mathcal{L}^1$. Only the simplest case will be considered, i.e. the subject adopts a plan at the outset and this plan remains invariant throughout the experiment. Let the information measure on the set of strategic alternatives ($\mathcal{L}^1$ statement sequences) be given as I_{max}. Since the plan is unchanged $dI_{max}/dt = 0$; the subject's $\mathcal{L}^1$ behaviour is, however, increasingly constrained as the plan is executed. Hence, his actual uncertainty about what $\mathcal{L}^1$ statements to make (say, I(t)), decreases with t and the $\mathcal{L}^1$ system is thus self-organising. Deviations

from this paradigm will be manifest as statements of "ℓ^1 doubt" and it is predicted that the number of probabilistic ℓ^1 statements will be less in the adaptive metasystem than in a free learning experiment (see 2.5.17).

Finally, one might reasonably expect to discover a consistent decrease in H or u as learning proceeds with respect to a given subproblem class, even if selection order is neglected. Such a behaviour would be indicative of successful self-organisation and any deviations from it suggest imperfections in the self-organising system. The data of Fig. 25 and Fig. 26 are obtained by lumping subproblem classes in this fashion and grossly indicate the superiority of the adaptive metasystem as a self-organising system.

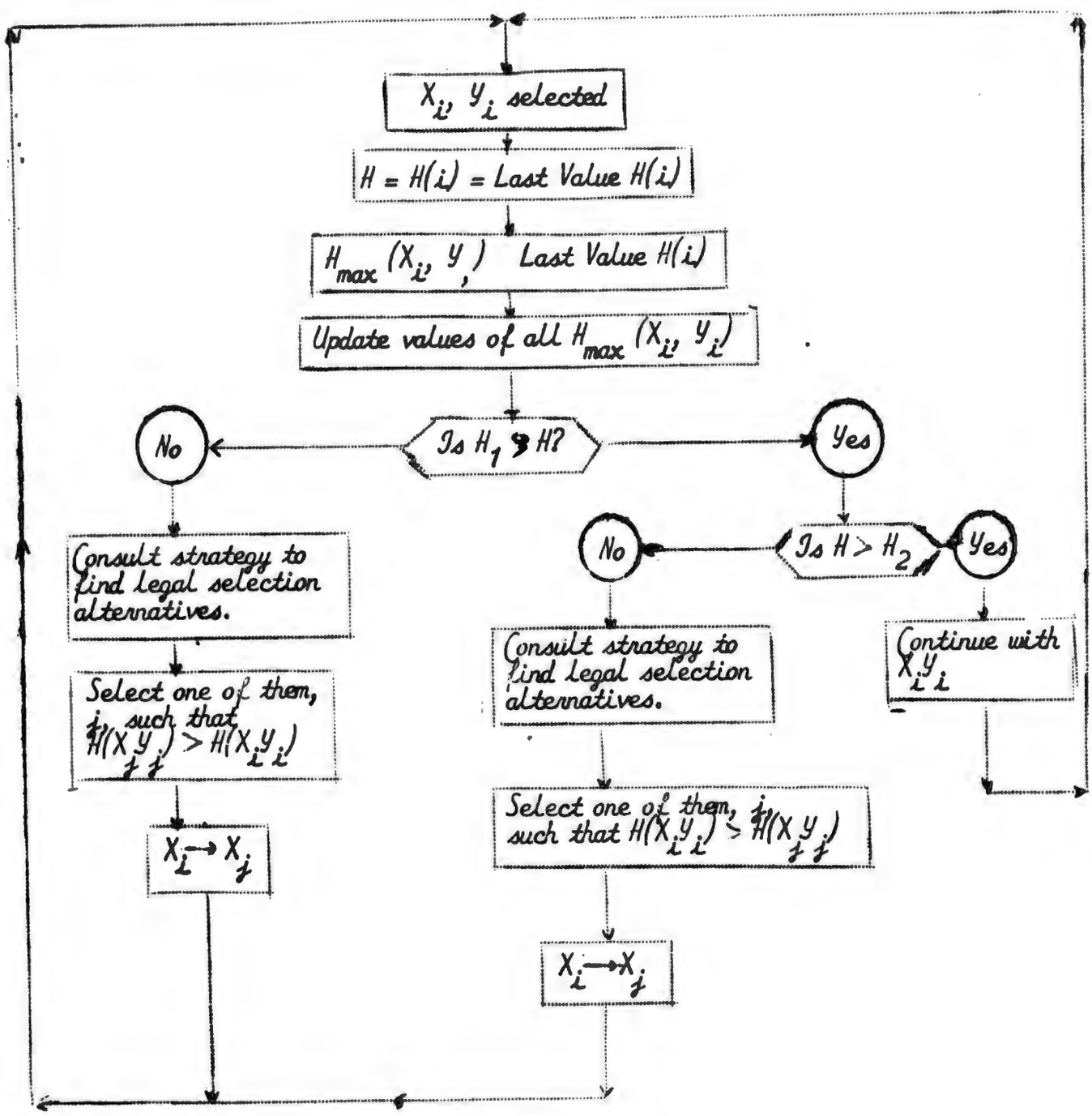


Fig. 27. Flow Chart for a Simple Self-Organising System.

SECTION : 4

PROFICIENCY MEASURES AND THE SELF-ORGANISING SYSTEM

4. Proficiency Measures and the Self-Organising System

In previous papers, (11) (12), Mallen has proposed a proficiency measure of the form $P = \frac{\rho\eta}{\eta_{max}}$ where η is an index over the levels of difficulty of the problems (uniquely stated in the case of a simple adaptive teaching system) and ρ is any performance measure of the type $p(y = \Omega(x) \mid x)$.*

Now, it is quite clearly possible to compute a P function in the present case. The act of doing so suggests an alternative (and for some purposes more acceptable view of self-organisation). It is as follows:

Let $H_{max}(t) = H^*(S_i)$ for S_i selected at t.

Now we do not propose to evaluate $H^*_{max}(S_i)$ but, simply, to rank order the $H_{max}(S_i)$, using E.(3). Let us call the level of a sub-problem class in the hierarchy $\eta = 1, 2, 3, 4$. From Eq.(3), $H^*_{max}(S_i) = f[\eta(i)]$ and the subject may be conceived as aiming for an increasingly high level of goal as he tackles problems indexed by a higher value of η. In the terminal condition, he is necessarily aiming for the highest level of goal, $\eta = 4$, and, consequently, if S is a self-organising system, then P should increase in a fairly regular fashion. For any subject pursuing a Type A strategy, it does so, and the plots for an adaptive meta-system (even with Type C strategy) show the same tendency.

FOOTNOTE: *This function is akin to the cost functions discussed by Cowan and myself(9) insofar as the teaching machine "gains" more by impelling the subject to perform successfully with respect to a difficult problem than it "gains" if he performs successfully with respect to an easy problem.

SECTION : 5

THE COMPUTER SIMULATED LEARNING MODEL

5. The Computer Simulated Learning Model

The real system described in Section 1 and Section 2 consists in a subject interacting with an automaton that controls the experimental conditions. In some of the experiments this automaton was no more than a procedural device; in others it was a simple adaptive teaching machine or the mechanical component of an adaptive metasystem.

The computer programmed model simulates the activity of the entire system. Hence, it contains a part which simulates the automaton (which is isomorphic with the automaton) and a part which simulates the subject or student. In the following discussion we shall take the automaton for granted, since the several varieties of automaton have already been described, and will concentrate upon the part of the model that simulates the learning process in the subject.

5.1. Broad Characteristics of the Model

(1) The model is an "artifical intelligence" programme (i.e. an organisational structure, not a stochastic model, such as a stimulus sampling model). But the artificial intelligence is psychologically oriented; it aims to mimic processes in the mind rather than logically elegant computation methods.

(2) The model is cybernetic insofar as the elementary units are goal directed (problem solving) systems, which may be represented as TOTE units, and insofar as these are put together to form larger goal directed systems.

(3) The model is heterarchical in the sense that its structure involves several separable, but interacting, hierarchical systems.
(4) The model is evolutionary. At any rate, this is true of the main part of the model, which, as below, simulates events in intermediate or working memory.
(5) The model is partially sequential. In this respect it mirrors the subject's propensity to do one thing at once, or at least to aim for one goal at once. On the other hand, there are parallel components; especially parts of the executive, which are programmed, like McCulloch and Kilmuirs' simulation of the reticular orienting system (13) to act as though they overlooked several possible modes of goal directed action at once. Further, there is a sense in which the model can do several things at once, i.e. it acts, in some ways, as a parallel processor. Of these features (1) calls for no further discussion. The rest are developed by way of an introduction to the programme itself, in the following sections.

5.2. The Stimulus Response and the Problem Solution Viewpoint

A human subject may nearly always[ø] be regarded as a Black Box in the sense of Ashby (15). This formulation calls for a minimal structural or organisational commitment. However, as soon as the picture is elaborated, several alternative methods of explanation become possible, and two of these have particular importance. (a) The system within the Black Box is conceived as a reactive device which responds to stimuli. (b) The system

---

FOOTNOTE: ø For practical purposes, always. But certain types of evolutionary behaviour(14) may provide exceptions to this rule.

is conceived as a problem solver which interprets λ expressions (amongst them stimuli), as inputs designating problems and which outputs responses which are λ expressions designating solutions. Alternative (b) will be adopted as the only one which is compatible with the psychological facts and the general notions of Cybernetics. But, in adopting it, we must, of course, provide a framework which justifies the title "interpreter" or "problem solver".

The paradigm case of a problem solver is the TOTE, or Test operate Test Exit unit of Miller Gallanter and Pribram (16), which is outlined in Fig.28. Perhaps the simplest realisation of the TOTE unit, applicable in the present discussion, is the general interpretative processor for context free languages, namely, the push down storage automaton of Fig.29; (in fact, the interpreter used in the learning model is somewhat more complex since it deals with contingent expressions, however, the automaton of Fig.29 is an interpretative problem solving device in the required sense).

Consider Fig.29. The automaton has the overall goal of solving the problem of making a well formed λ expression designating a solution to the problem posed by any stimulus, x. First, a stimulus (either simple or complex) is interpreted as posing a problem. There is an unordered set of rewriting rules. The left hand side (domain) of any rule is the symbol for a stimulus, or a part of a stimulus. The right hand side (range) of the rule is either the name for another symbol (in which case the rewriting rule is a decomposition rule), or the name for a response programme (in which case the rewriting rule is an operator rule). As in Fig.29, the automaton solves problems insofar as it computes, until all of the symbols have been converted into response programme names, and until the named response programmes have been executed.

Further, it acts precisely as a TOTE unit since it continually asks, "is the problem solved" (or "is the goal achieved") and, if not, it operates by applying a rewriting rule. Insofar as the computation is successful, it generally consists in reducing the original problem to subproblems, which are solved in sequence. But, clearly, the computation will not always be successful unless the automaton has an adequate set of rewriting rules. If success is not achieved, the automaton invokes an auxilliary (learning) device to build up fresh rules so that the original problems may be solved; in particular, it builds up fresh operator rules, (which, in the learning model, are embodied in entities called $\mathcal{L}^0$ operators).

A word of caution is in order at this point. Taken in isolation, an $\mathcal{L}^0$ operator rule looks very much like one of the associative bonds which connect stimuli to responses in a reactive model (an alternative (a) model). Hence, the accretion of new rules, by dint of learning, superficially resembles "building up associative connections between stimuli and responses". However, I wish to maintain that an operator rule is quite <u>meaningless</u> when taken alone. It has meaning if, and only if, it is interpreted in the <u>context</u> of a processor, such as the device in Fig.29. If it is so interpreted, then the act of building up a collection of operator rules is isomorphic with the act of building up one or more TOTE units. This is <u>one</u> sense, the <u>simplest</u> sense, in which the learning model builds up problem solving programmes, any one of which is a TOTE unit.

5.3. <u>The Heterarchical Character of Learning</u>

In a comprehensive study of human learning[✓], it is necessary to state and distinguish between several hierarchical systems.

FOOTNOTE: ✓ Learning to solve problems, concept learning, etc. Probably this is also true for all sorts of goal directed adaptation.

5.3.1. The Hierarchy of Storage Systems

It is well known that there are several functionally distinct types of information storage involved in most human learning processes, and that these form an hierarchy in terms of their time constants or decay times. Each system must be represented in a competent learning model.

First, there is immediate storage with a capacity measured by the span of apprehension.✓ This is a buffer store, which may be easily overloaded, but in which elements retain their integrity. Symbols are not mutilated and do not interact with one another. Immediate storage may be organised into many different memory systems (for example, as a rehearsal buffer, or as a push down store). In the processor of Fig.29, the "problem" and the "solution" are both written in immediate memory and, if the storage capacity is exceeded, something will be lost.

Next, there is intermediate or working storage. It has an indefinite capacity, which is chiefly limited by the decay of representations and interference between the stored items. Intermediate memory, the organisation of intermediate storage, is the seat of interference, positive transfer and, probably, of consciousness also. In the processor of Fig.29, the operator rules are stored in this system; so, probably, is the organisation of the processor. In general, programmes that operate upon data in immediate memory are stored in the intermediate memory system (it can, indeed, be argued that immediate memory is a specially organised piece of the intermediate store. If so, its organisation protects the immediate store contents from mutilation and abrasion).

FOOTNOTE:

✓ We are neglecting the visual short term buffer and the other perceptually specific storage locations; in other words, immediate storage is really immediate symbolic memory.

Finally, there is a virtually inielible long term storage system of very high capacity, but largely unieterminei accessing properties. As a memory, I believe it is functionally iientical with a system of Von Foerster's cognitive tiles (17). In the processor of Fig.29, a representation of the iecomposition rules is implanted in long term storage ani, likewise, a representation of the entire push down processor. However, as a matter of convention, it will be maintainei that these representations are copiei, non iestructively, from long term storage into intermeiiate storage (where they are usei operationally ani may even be moiifiei). Further, after utilisation, the possibly moiifiei representation may be copiei back ani aiiei to the contents of long term storage.

Hence, there is an hierarchy of the form:

Long Term Memory

Intermeiiate or Working Memory

Immeiiate Memory

5.3.2. The Hierarchy of Goals

There is an hierarchy of goals in which a main goal is at a higher level than a subgoal. The most explicit form of this hierarchy is a plan formei, a priori, of TOTE units (as in Fig.30). Formally, this construction is obtainei by iecomposing the operational phase of a TOTE unit with goal G, ani replacing it by a pair (or more) of TOTE units with subgoals, G_1 ani G_2. In the main learning moiel, plans of this type are callei λ^0 operator strings.

On the other hani, subgoals can be generatei, as we saw in Fig. 29, from the operation of a single TOTE unit.

FOOTNOTE: There is a slight ambiguity in the notion of a TOTE unit since it may either be interpretei as an iterative or a recursive proceiure. In the latter case only, can there be subgoals proiucei by the operation of a single TOTE unit.

5.3.3. The Hierarchy of Operator Order

There is an hierarchy engendered by the order of an operator rule, or strictly, by the order of the domain and the range of an operator rule. To illustrate this point, let $X_A X_B X_C X_D = X_1$. If $x_1 \in X_1$, then x is of the 1st order. An operator rule with x on its left hand side and the name for some single response component on its right hand side, is an operator rule of the 1st order. Let i, j = A, B, C, D. Clearly, $X_{ij}^{\cup} \in X_i \times X_j$, so that any stimulus of the form, $x_2 \in X_2 = \bigcup_{ij} X_{ij}$, is an ordered pair and is of the 2nd order. Similarly, an operator rule with x_2 on its left hand side and (in the limited environment of Section 1 and Section 2) the name for an ordered pair of response components on its right hand side, is an operator rule of the 2nd order. In general, there are operator rules of order $\eta = 1, 2, 3, 4$ (relevant within the universe of discourse of Section 1 and Section 2) and in the main learning model these are embodied in $\mathcal{L}^0$ operators of order η.

5.3.4. The Hierarchy of Control Systems

There is an hierarchy of control with levels, L^0, L^1,..... Problem solving programmes (in general, operators in the context of an interpretative device) may reside at any level in this hierarchy. The L^0 programmes (essentially the only ones we have so far discussed) solve problems posed by stimuli from the environment; their domain is in the stimulus set and their range is in a set of response programme names. In contrast, programmes at level L^1, have a domain of properties of the L^0 repertoire) and their range is a set of new L^0 procedures; for example, of new rewriting rules. Hence, L^1 programmes construct or modify L^0 programmes. They are responsible for learning. The overall hierarchical organisation is identical with the adaptive control systems discussed by Tarjan and is outlined in Fig.31. It should be emphasised that an L^0 problem solving programme is logically the same sort of entity as an L^1 problem solving programme and that the difference

between the two lies in the domain upon which the programme operates (L^0 programmes solve problems posed by the environment. L^1 programmes solve problems due to the current deficiency of the L^0 repertoire)*. The hierarchy of control is in one-to-one correspondence with the hierarchy of levels, $\mathcal{L}^0$, $\mathcal{L}^1$,.... in the object language, $\mathcal{L}$. Thus, any $\mathcal{L}^0$ statement is interpreted by an L^0 programme, and any $\mathcal{L}^1$ statement is interpreted by an L^1 programme.

5.3.5. The Interdependence of the Hierarchies

It is important to appreciate that these hierarchical organisations are largely interdependent. Thus, an hierarchy of goals can exist at any level in a control hierarchy, i.e. there may be more or less abstract operational plans as well as more or less abstract learning strategies. By the same token, operators of different orders can exist at the same level in a goal hierarchy. Finally, more or less abstract goals and problem solving programmes, at higher and lower levels of control, may be lodged in immediate, intermediate and long term memory. Thus, the learning model is heterarchical rather than hierarchical, as maintained at the outset.

5.4. Overall Organisation of the Simulation

The overall organisation is shown in Programme Flow Chart, (PFC) 1. Each trial is divided into two phases (1) a phase lasting for δt secs. at the beginning of which the simulated automaton delivers a problem posing stimulus designating a problem in some (already determined class). In reply, the learning model outputs a (possibly

---

FOOTNOTE: *As an alternative construction, the hierarchy of control can be envisaged as an hierarchy of reproductive systems. Thus, L^0 problem solving programmes are automata (any TOTE unit can be equivalently represented as a finite automaton). The cannonical form of an L^1 problem solving programme, is an automaton for reproducing L^0 automata (possibly with variations). The cogency of this viewpoint will become apparent in our discussion of the main learning model.

vacuous) response, the components of which are associated with latency values (2) a phase lasting $\Delta t - \delta t$ secs. in which the simulated automaton delivers either partial or complete knowledge of results, and the learning model interprets and processes this information.

At the end of a block (determined by the simulated automaton), the programme is temporarily halted. In case free learning is being simulated, there are two possibilites (I) The programme outputs a "demand" statement via the teletypewriter, together with the behavioural data from the previous block (including ρ and the last values of ρ). The experimenter replies to the "demand" by typing the next subproblem class for attention and (if required) new values of subject parameters. (II) Control is assigned to a subprogramme simulating the subject's internal attention directing mechanism, which automatically makes the requisite selection (and, if required, parameter variation). In case a teaching situation is being simulated, control is transferred to the simulated teaching machine (serving as an external attention directing mechanism) and this device makes the necessary decisions. Apart from the decision rules involved, the "internal" attention directing subprogramme and the "external" teaching machine programme have an identical status.

5.5. Organisation of the Learning Model

The learning model is made up from the following components:

(1) A representation system in which stimuli, responses, knowledge of results signals and so on are given names (literally numbers).

(2) An L^0 interpretative processor called the application procedure. The application procedure is slightly more elaborate than the push down list processor of Fig.29 but, like this device, it works by applying rewriting rules (it also applies ordered sequences of rewriting rules). Any rewriting rule is embodied in an entity called an L^0 operator, and any ordered sequence of rewriting rules is embodied in an L^0 operator string.

(3) A system called "working storage", which simulates the subject's intermediate memory (immediate memory is simulated by certain list structures in the application procedure). All L^0 operators and L^0 operator strings are lodged in working storage and the collection available (to the application procedure) at a given trial, is called the L^0 repertoire at this trial. Since an L^0 operator, or operator string, is a TOTE unit when taken in the context of the application procedure, the L^0 repertoire is a repertoire of L^0 problem solving programmes.

(4) A number of L^1 problem solving programmes, which build up, modify and preserve the L^0 repertoire in working storage. These are: (I) Operator Creation (with a subroutine called "guessing").

(II) String Creation (by the concatenation of a pair of operators to form an operator string).

(III) String Extension in which an operator (or a string) is concatenated to an existing string.

(IV) Substitution of an operator string (to form a complex operator of order greater than 1).

(V) Deletion (of specific operators).

(VI) Reproduction (which, as below, is necessary to maintain an L^0 operator, or operator string in working storage).

(VII) Attend to knowledge of results.

(VIII) Make an analagous programme or analogy (this procedure is so far only partially realised).

(5) An overall executive. By hypothesis, the subject has a limited amount of mental effort available over a given interval. To mirror this postulate, the model is given a limited amount of computational effort over a given interval. The

executive allocates this limited supply of effort to various processes (which, in an obvious sense, compete for the effort available). All of the procedures so far mentioned require the expenditure of effort if they are run, i.e. L^0 Application, L^1 Creation, L^1 Concatenation, L^1 Substitution, L^1 Deletion, L^1 Reproduction and L^1 analogy. In other words, each of these processes has a certain cost measurable in the units of effort required for its execution, and each process is assigned a proportion of the available effort.

5.6. Representation

The representation scheme is shown in Fig.32 and is largely self explanatory so far as L^0 statements are concerned. The $\mathcal{L}^1$ constraints imposed upon the subject when he accepts the experimental contract and the descriptive scheme of Section 1, are built into the programme and cannot be easily modified. Thus, the stimuli are numbered as decompositions of 4 lamp stimuli; if the SN (stimulus number) of x is less than 9, for i, j = A, B, C, D, then $x \varepsilon X_1 = \bigcup_i X_{ij}$, if $32 \geqslant SN(x) > 9$, then $x \varepsilon X_2 = \bigcup_{ij}(X_i, X_j)$, if $64 \geqslant SN(x) > 33$, then $x \varepsilon X_3 = \bigcup_{ijk}(X_i, X_j, X_k)$, and if $72 \geqslant SN(x) > 64$, then $x \varepsilon X_4 = \bigcup_{ijkl}(X_i, X_j, X_k, X_l)$.

Similarly, the application procedure embodies the notion that Ω is a one-to-one correspondence.

The basic $\mathcal{L}^1$ statements of knowledge of results (either partial or complete), of ρ, and of last value ρ, are provided automatically.

Other $\mathcal{L}^1$ statements involve changes in subject parameter (SU) values, which are, however, easily manipulated either by the experimenter or the (simulated) controlling automaton in the course of an experiment. For example, one subject parameter (SU[3]) is a threshold on string utilisation, which must be exceeded before that string can be substituted, and the experimental statement, "try to see problems as a whole" (which is issued, in an adaptive metasystem, before the grouping test) is imaged by setting SU[5] to its minimum permitted value.

5.7. L^0 Operators, Operator Strings and the Working Storage

An L^0 operator is a word containing the following information,

Utility	Lifespan	Domain	Range	Name

where the domain is the name of a stimulus and the range is the name of a response programme. If the domain and range are of order 1, then the L^0 operator is simple; if their order is greater than 1, then the operator is complex. In either case, the operator becomes a TOTE unit in the context of the application procedure, and in that context, the stimulus poses a problem. Utility and Lifespan are variables, which will be discussed in connection with the working storage system.

An L^0 operator string is a similar word. Its domain is the name of a stimulus of order greater than 1 (if its order were 1, the string would be a trivial string of length 1). But unlike the L^0 operator, its range is an ordered collection of names for L^0 operators (to be searched for and later applied). Hence, an L^0 operator string contains the following information -

Utility	Lifespan	Domain	Description List of Names of L^0 Operators	Name

In the context of the application procedure, an L^0 operator string becomes a nested sequence of TOTE units, i.e. a structure of the type shown in Fig.30.

Working storage holds L^0 operators and L^0 operator strings. An L^0 operator may first enter working storage (I) as a result of operator creation or (II) as a result of substitution. An L^0 operator string may first enter working storage (I) as a result of string creation, or (II) as a result of string extension.* When it

---

FOOTNOTE: *Either type of entity can first enter working storage as the result of an analogy operation. However, analogy is not flow charted in this account of the system.

first enters working storage, an operator (operator string) is assigned an initial value of lifespan. Subsequently, it is subject to decay, insofar as the value of lifespan is decremented at each trial by an amount proportional to the number of members of the L^0 repertoire. If the lifespan variable of an operator (operator string) reaches a value of zero, then the entity concerned is deleted from working storage. Hence, in the absence of any other process, all operators and operator strings would inevitably disappear from working storage at a definite number of trials after their first entry.

This fate is avoided (and some entities do survive) because of L^1 reproduction.* The reproductive process picks out the name of an operator (operator string) from an "awaiting reproduction" stack in which a single name may appear more than once. When a name is picked out, the lifespan of the corresponding operator (operator string) is given its initial value (clearly, this is equivalent to reproducing or recreating the entity in question) and shortly afterwards the picked out occurrence of the name is deleted from the "awaiting reproduction" stack.

Now, each operator (operator string) is also associated with a utility variable. At first entry into working storage, utility is assigned a small positive value. Subsequently, the value of utility is increased whenever the entity concerned is used as part of a successful L^0 problem solving process, i.e. whenever this L^0 entity is employed by the application procedure in connection with a problem and where an affirmative knowledge of results signal is obtained. Otherwise, the utility value is reduced, at each trial, by a constant decrement. If, at any trial, the value of utility for a given operator (operator string) exceeds some constant threshold, the name of the entity is entered in the "awaiting reproduction" stack.

FOOTNOTE: *In certain other publications called L^1 maintenance.

Reproduction involves the expenditure of effort (the reproduction cost) and the amount of effort available for reproduction, at a given trial, depends upon the total effort and the assignment (to reproduction) made by the executive. The number of items in the awaiting reproduction stack that are actually processed at trial n is, thus :

$$\frac{\text{Effort Available for Reproduction at Trial n}}{\text{Cost of Reproduction}}$$

However, given this quantity, the chance that a particular operator (operator string) will be reproduced at trial n, depends upon the frequency with which its name appears in the "awaiting reproduction" stack and this, as above, depends directly upon the number of previous occasions upon which the pertinent utility value has exceeded threshold.

The working storage system thus involves competition amongst operators (operator strings) for reproduction and survival. In general, not all of the operators (operator strings) can survive because of the limited effort which is available for reproduction; Hence, the working storage system is governed by a selection principle. Operators (operator strings) that are not reproduced are generally inadequate (fail to elicit affirmative knowledge of results signals) and are replaced by putative variants. The source of variety is twofold (I) it stems from exploration of the problem classes in the environment (II) it is provided by an inbuilt guessing procedure (part of the creation process), which selects alternatives randomly within constraints determined by the prevailing stimulus. Since the working storage system is based upon this combination of competition, reproduction, variation and selection, it is, by definition, an evolutionary system.

5.8. L^0 Application Procedure

The fundamental L^0 processor is the application procedure. The procedure may come into play after stimulus presentation during phase 1 of P.F.C.1. It should be stressed that application need not occur after the presentation of a stimulus (though it usually does so). The real subject may refrain from trying to solve problems for a few trials (whilst he works out how to solve them, for example). Precisely the same comment applies to the learning model. Whether it does apply at a given trial, is determined by the executive. However, on the assumption that application is attempted at trial n, the entire process of P.F.C.2. (application procedure) is carried through without interruption, i.e. application is conceived as an autonomous process, which, once triggered, is carried to completion. However, it is entirely possible (I) that the procedure yields no response or (II) that some, or all, of the response component latencies exceed δt. In either of these cases, the response is deemed to be mistaken.

At the n^{th} trial, x(n) is selected by the automaton; as in P.F.C.2., the stimulus number, SN(n) of x(n), is placed in immediate memory. The working store is examined; if it contains any L^0 operators, (i.e. if the operator store is not empty) the domains of the existing operators are matched against the stimulus number and a test for matching is carried out (successful if, and only if, a match is obtained). If so, then there is an L^0 operator with a name O(n), such that SN(n) = Domain $[O(n)]$. In that case, O(n) is applied. The range of O(n) is retrieved (by definition, a response programme name) and in the case of a complex operator of order > 1, the name for a group of response programmes). The programme is (these programmes are) executed in the manner indicated.*

FOOTNOTE: * We return to the details of response programme execution in a moment.

If the matching test is unsuccessful, or if no L^0 operators exist, the programme tests for $9 \rightarrow SN(n)$ (that is, for $x(n) \in X_1$). If the test result is affirmatively valued, then a simple operator (order = 1) is required and (from the searches so far conducted) is not available. Consequently, the programme attempts to construct an operator, flow charted in P.F.C.3. In case the test yields a negative result, the working store is searched for the names of L^0 operator strings. Supposing no strings exist (string store empty), the programme either tries to create an L^0 operator string, using L^1 string creation (flow charted in P.F.C.4., P.F.C.5., P.F.C.6., for $x(n) \in X_2$, $x(n) \in X_3$ $x(n) \in X_4$), or to create an L^0 operator (P.F.C.3), depending upon whether or not there are any L^0 operators in the working storage. If, on the other hand, some L^0 operator strings do exist in the working storage, the programme searches their domains in an attempt to match SN(n) to one of them. Suppose no match is achieved, control is turned over to P.F.C.4, P.F.C.5, or P.F.C.6; suppose a match is obtained, the string name, S(n), is retrieved.

At this point, S(n) is applied. First, there is a sequential search of the range of S(n), which, by definition, contains the names of several response programmes. For each name encountered, an attempt is made to retrieve the corresponding L^0 operator from working storage; clearly, this operator may, or may not, exist (it must have existed at some stage but it may have been deleted before trial n). If the search is successful, the operator is applied, leading to the execution of a response programme, as before; if not, the missing operator is marked. The search through the range of S(n) is continued until all of its components have been examined.

The execution of one or more response programmes, either resulting from L^0 operator application or from L^0 operator string application, gives rise to a response vector y(n), which is represented as a Keys Pressed List, KP. It should be emphasised that y(n) (the vector, KP) is not necessarily the correct response, i.e. although

the L^0 operators or operator strings, which yield y(n), have x(n) in their domain, there is no guarantee that they are the correct operators.

Two further processes (suppressed in these notes) are actually carried out by the programme of P.F.C.2. (I) A List, A, is formed, containing all of the operators applied at trial n and a list, D, is formed containing all of the operator strings applied at trial n. (II) Each component of y(n) (or KP) is associated with a latency composed of several parts, as detailed in the flow chart.

If the exit from P.F.C.2 is through an operator application or an operator string application (rather than through P.F.C.3, or P.F.C.4, P.F.C.5, P.F.C.6 control is transferred to the automaton simulation at the start of phase 2 in P.F.C.1 (i.e. comparison between y(n) and $\Omega[x(n)]$ and comparison between the latencies and δt; if any latency exceeds δt, the associated response is deemed to be mistaken).

5.9. L^1 Operator Creation

This routine is flow charted in P.F.C.3. It is entered from P.F.C.2. (as described already) or from various points in the string construction and extension routines, P.F.C.4.,P.F.C. 5. and P.F.C.6, or from P.F.C.8. At the moment of entry, some stimulus number, SN at the nth trial, is available in immediate memory. If it happens that SN is 8 or less (as in exit 1 from P.F.C.2) then $x(n) \varepsilon X_1$, and an operator of order 1 is required. This is obtained by writing SN as the domain of a new operator, which is assigned a new name in working storage. The range of the operator is written by guessing, randomly, amongst a subset of the single response components; initially, this subset consists in all 8 alternatives; later its size is reduced by deleting previously guessed alternatives.

The lifespan and utility of the new operator are given predetermined initial values (specified by subject parameters, SU). Immediately afterwards, the operator is applied (as in P.F.C.2).

If SN > 8, then $x(n) \varepsilon X_\eta$, $\eta > 1$ and the problem is decomposed to its order 1 constituents in the sense that rewriting rules are applied to express it in terms of η separate constituents $x^* \varepsilon X_1$. This procedure yields η separate decomposed stimulus numbers $SN\uparrow_i^*$. Each of the stimulus numbers is taken in turn as the domain of an operator of order 1, which is created and applied as above. In other words, the learning model tries to solve the problem part by part.

L^1 operator creation may be interrupted by the executive at any step. If it is interrupted (and it will be if the effort assigned to creation runs out) control passes to the automaton simulation at the start of phase 2 of P.F.C.1.

5.10. Problem Decomposition

We have already come across one instance in which it was necessary to apply a decomposition rule to the problem designated

FOOTNOTE: *See Section 5.10. for comments on this notation.

by a stimulus, i.e. in P.F.C.3. to obtain order 1 decompositions of x(n). In general, decomposition, takes place before any of the string construction routines in P.F.C.4, P.F.C.5, or P.F.C.6 and these are written on the assumption that the decompositions of the original problem are available in immediate memory. Now, two assumptions are possible at this juncture, namely (1) that decomposition is unrestricted so that, for example, there are 15 decompositions of a problem in class X_{ABCD} (including the original problem). (2) That a restricted set of decomposition rules are applied, for example, ABCD→ABC, D: ABC→AB, C: AB→A, B where the freedom of selection is successively reduced (i.e. conceivably there are 4 "single-triple" order 3 decompositions of ABCD; once that one of them is selected, there are 3 "double-single" order 2 decompositions of the triple component, alternatively, there are 6 order 2 decompositions of ABCD, and so on. Of these assumptions, the first is the least realistic but is used to exhibit the string construction and string extension processes.

Explicitly, x(n) is decomposible into a subset of product n-tuples (typically, $\langle x_1, x_j \rangle$) such that $\langle x_1, x_j \rangle = x(n)$, where the stimuli concerned are members of subsets $\{x_1\} \subset X_{\eta_1}$ and $\{x_j\} \subset X_{\eta_2}$ (for example, if $x(n) \in X_{ABCD}$, then it is decomposable into:

$\langle x_1, x_2 \rangle$ where $x_1 \in X_{AB}$, $x_2 \in X_{CD}$ or into $\langle x_2, x_1 \rangle$ or into:

$\langle x_3, x_4 \rangle$ where $x_3 \in X_{BC}$, $x_2 \in X_{AD}$ or into $\langle x_4, x_3 \rangle$ or into:

$\langle x_5, x_6 \rangle$ where $x_5 \in X_{AC}$, $x_6 \in X_{BD}$ or into $\langle x_6, x_5 \rangle$ here

$\eta 1 = \eta_2 = 2$).

Now each of these decompositions corresponds to a numbered item in Fig.32. Hence, given SN designating x(n), we employ the notation

SN → m complementary n-tuples $\langle SN_{\eta_1}\ i,\ SN_{\eta_2}\ j \rangle$

to represent the rewriting process that gives rise to the numbered decompositions.

In order to match the domain, some operator (or string) against some first member of a decomposition, it is necessary to repeat the original "Domain Search" matching process for each possible value of i until a match is achieved. The requisite subroutines ("Find operator" and "Find string") are shown in P.F.C.4.

5.11. L^1 String Creation for $x(n) \varepsilon X_2$

The process is outlined in P.F.C.4. Since the minimal non trivial string has length 2, there is no question of extending a string to length 2; hence, for $x \varepsilon X_2 (33 > SN > 8)$ only string creation is possible. If the operator store is empty, the operator creation (P.F.C.3.) and application (P.F.C. 2.) routines are applied to the 1st and the 2nd member of the order 1 decomposition of x(n) (of SN) and control is returned to the subroutine, "create string". As a result of double operator application, there will be, at this stage, 2 operators in list A (of P.F.C.2.)

"Create String" consists in assigning a name to the new string, writing x(n) (SN) in its domain, writing the name of an operator applicable to the 1st member of the decomposition of x(n) and another applicable to the 2nd member of the decomposition of x(n) in its range and giving initial values to the lifespan and utility variables. The resulting string is, thus, entered in working storage.

If the operator store is not empty, one or both of the requisite operators may be discovered in existence. The remainder of P.F.C.4. deals with the contingencies that may be encountered (both operators exist, only one of them exists) before reaching the terminal subroutine, "make string".

After "make string" has been executed, control is transferred to the beginning of phase 2 of P.F.C.1. As in the case of L^1 operator creation (or any other L^1 procedure), string creation may be interrupted by the executive. If it is interrupted, control is transferred to the beginning of phase 2.

5.12. L^1 String Creation or Extension for $x(n) \varepsilon X_3$

The process is shown in P.F.C.5. For $x(n) \varepsilon X_3$ (i.e. 64→SN →33) there is a priori, a possibility of creating a new string or of extending any string that exists. Whether the model attempts to do one or the other depends upon the value of a subject (SU) parameter and a random choice. The rest of P.F.C.5. is self explanatory, given the following comments -

(a) The subroutine "extend string", given a string $S(n)$, consists in adding the name of a further operator to the range of $S(n)$.

(b) Whereas, in P.F.C.4., it was only necessary to consider operators of order 1 (and decompositions in X_1), the present process is generalised to operators of order 2, as well.

(c) The complement of a decomposition x_i ($SN\eta_1 i$) of $x(n)$ is a further element of x_j ($SN\eta_2 j$) such that $\langle x_i, x_j \rangle = x(n)$.

After the execution of either "String Extend" or "Create String", control is transferred to the beginning of phase 2 (of P.F.C.1.). Control is also transferred to this point if the process is interrupted by the executive.

5.13. L^1 String Creation, String of String Creation, and String Extension for $x(n) \varepsilon X_4$

In case 72 SN ≥ 64, $x(n) \varepsilon X_4$ and the string construction process is shown in P.F.C.6. The possible decompositions and string structures are clearly more complex than those previously considered but the only essentially novel feature introduced is the string of strings, that is, a string with the names of other strings (rather than the names of operators) in its range. Such an entity is built when a domain search for a matching operator is unsuccessful, but when there are two or more (with $\eta_{max} = 4$, two at the most) strings in working storage with domains that jointly cover the problem.

The system, P.F.C.6., may be interrupted by the executive. If it is interrupted, control is transferred to phase 2 of P.F.C.1.

5.14. Response Measures (Automaton Simulation)

Phase 1 (of P.F.C.1.) terminates with a vector, KP, which represents the keys pressed by the subject at the nth trial, i.e. y(n), and a further vector of latencies for each component of the response. Notably, entries do not appear in KP if the latency of the corresponding response component exceeds δ t. SN, representing the problem posing stimulus, x(n), is also available at this juncture. The automaton simulation forms the order 1 decomposition of SN (i.e. of x(n)) and represents this decomposition as a vector, DL; it forms a further vector, RL $= \Omega[DL]$ (i.e. the representation of $\Omega[x(n)]$) and it matches RL against KP for equality. If KP = RL, then a variable, OK, is set to a value of 1 ($\sigma_o = 1$); if KP $\neq$ RL, then OK = 0 ($\sigma_o = 0$). The automaton also forms a vector, DK, (detailed knowledge of results) such that DK[I] = +1, if KP [I] is a correct component of the response and DK [I] = -1 if KP [I] is not a correct component of the response (here I is the index of the components in the vector, KP).

5.15. Processing the Knowleige of Results Information

This part of the learning modal programme is executei at the start of Phase 2 in P.F.C.1. and is shown in P.F.C.7(I) and P.F.C.7 (II) Of these flow charts, P.F.C.7(I) is concernei with partial knowleige of results and P.F.C.7(II) with the ietailei knowleige of results.

If OK = 1 (i.e. if $\sigma_o = 1$), all operator utilities on list A are incremented and if any string appears on list D, its utility is also incremented. If OK = 0, and ietailei knowleige of results is available (in the condition being usei), control is transferrei to P.F.C.7(II). If ietailei knowleige of results is not available, the system ietermines whether a unique operator was appliei (if so, it must be mistaken and its name is aiiei to a Delete Stack). A similar enquiry is maie regarding a unique string. After this, control is transferrei to either the substitution routine of P.F.C. 8. or the reproiuction routines of P.F.C.9.

In the case when ietailei knowleige of results is provii ei and OK = 0, the following process takes place (inaicatei in P.F.C.7(II)). DL and RL are loigei in immeiiate memory. For processing convenience, list A is copiei into a working list, B, and list D into a working list, C, the last operator appliei being the leftmost member of B (similaraly for strings in C).

The first member of B is examineil. Its domain is ietermineil and the 1st orier iecomposition of its domain is written in a list, TS. Its range (and, thence, the keys grassei) is ietermineil and the key numbers are copiei into a temporary keys list, TK. If TK is empty, then the operator is necessarily mistaken when TKR (temporary knowleige of results) = -1. Otherwise, the system makes the hypothesis that it is correct so that TKR is (tentatively) assigneil the value of +1. Now the members of TK are iniexei by a number, Q, and the correctness hypothesis is revisei (so that TKR→-1) if and only if DK [TK (Q)] = -1. The system next tests the value of the variable, TKR. If TKR = +1, then the 1st orier iecomposition of

the domain of the operator (i.e., TS) is deleted from DL and the keys pressed by the operator are deleted from RL. If TKR = -1, then the operator is added to the Delete Stack. At this point, the first operator is also deleted from list B. The next operator in list B is examined and the process is repeated, providing there are any operators remaining in B.

The system now goes on to process any operator string that may exist in list C (using only the OK variables as feedback).

Having completed these operations, there may or may not be entries left in DL and RL. If so, they have not been associated with a relevant operator. Hence, control is transferred to the operator creation subroutine in P.F.C.3. to create an operator with a domain consisting in the entries remaining in DL and with range, the entries remaining in RL. After that, control is transferred either to substitution (P.F.C.8.) or reproduction (P.F.C.9.)

The detailed knowledge of results procedure may be interrupted by the executive; if it is, control is transferred to substitution or reproduction

5.66. L^1 Substitution Process

The organisation of this process is shown in P.F.C.8. and it takes place during phase 2 of P.F.C.1. As a preamble, there exists a finite sized stack called the substitution stack, which contains the names of operator strings. Entry of a name into this stack (as in P.F.C.11.) is determined by a threshold test on string utility. In this respect, entry into the substitution stack is analgous to entry into the reproduction stack (outlined in 5.7. and flow charted in P.F.C.11). Effort is allocated by the executive to the act of substitution and the amount of effort determines the number of strings in the substitution stack that will actually be substituted at a given trial.

Returning to P.F.C.8., the first name in the substitution stack is picked out. Working storage is searched for the named string, say S* (it may have been deleted, though at some stage, S* must have existed). If S* is discovered, the working storage is searched for

the operators named in the range of S*. When an operator is found, the names of the responses which would be produced by its execution are recorded in a list, L. If all of the operator names in the range of S* can be found, L contains the names of all the responses engendered by the application of S*. (Some of these operators may, however, have been deleted and, if so, the procedure terminates). The substitution process now forms a newly named response programme, say R, such that, when executed, R produces all of the responses named in L. R is entered in the response programme list and a newly named operator, say O*, is formed and placed in working storage with domain the domain of S*, with Range R, and with initial values of lifespan and utility. Unless the substitution stack is empty, the process of P.F.C.8. is repeated until it is interrupted by the executive.

5.17. L^1 Reproduction Process

Like L^1 substitution, the L^1 reproduction process P.F.C.9 (already considered in 5.7.) is based upon a finite stack of operator or string names. The process of P.F.C.7. is carried out on this stack until either (1) the stack is empty or (2) the process is interrupted by the executive.

5.18. Deletion of Operators and Strings

The deletion of operators and strings may be an active process (in contrast to the deletion of operators having a lifespan of 0). A stack of the names of "mistaken" entities awaiting deletion was built up in the execution of P.F.C.7. L^1 deletion goes through thisstack, finds the entity in working storage and deletes it (as in P.F.C.10.) until either (1) the stack is empty or (2) deletion is interrupted by the executive.

5.19. Updating Procedures

A number of updating procedures are carried on at the end of phase 2 in P.F.C.1. These are not subject to interruption by the executive and it is assumed that they do not require effort. They are shown in P.F.C.11, and consist in (1) adding names to the substitution stack, (2) adding names to the reproduction stack, (3) updating the lifespan of all entities in working storage, (4) discarding those entities having a lifespan of 0 (in contrast to the active deletion of mistaken operators in P.F.C.10).

5.20. The Executive

With the exceptions listed below, the main processes in the learning model are simulated as though they were run in parallel and the executive is a supervisor which assigns computational effort to the several parallel processes. The exceptions are as follows:

(1) Application and string construction must occur in phase 1 (if they occur at all) and application must precede string construction (if it occurs).

(2) The knowledge of results processing must occur in phase 2 (if it occurs) and, consequently, it is preceded by application and string construction or extension.

To simulate parallel processing, the programme is provided with 5 clocks (variables CL(I)), I being the process and clock index, with limits INT (I). The clocks govern the 5 process groups:

A = Application and construction or extension (P.F.C. 2, 3, 4, 5, 6)

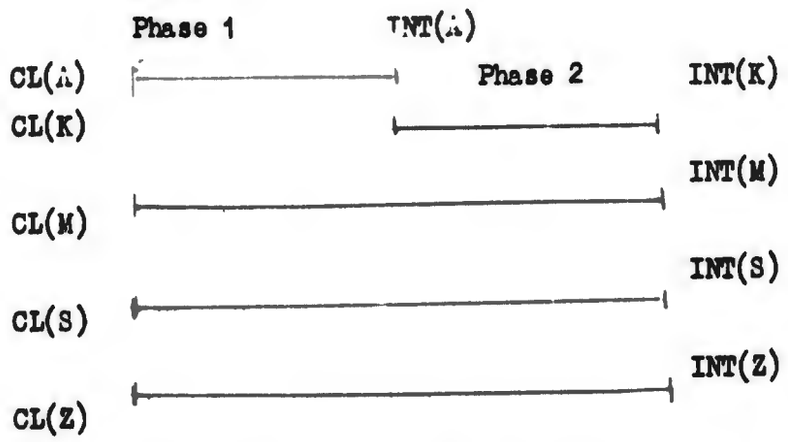
K = Knowledge of Results Processing (P.F.C.7,)

M = Reproduction (previously called maintenance)P.F.C.9)

S = Substitution (P.F.C.8)

Z = Deletion (P.F.C.10)

A pictorial representation is given below, INT(A) being set equal to δt, INT(K) being set equal to $\Delta t - \delta t$ and the remaining limits INT(M), INT(S) and INT(Z) being set equal to Δt.



Whenever a main process I is initiated, CL(I) is incremented by an amount, DCL(I), for each step in the programme (for each execution of a subroutine). It is possible that the process will terminate (there being nothing left to do) before CL(I) = INT(I) in which case an amount of "time" is left unused. However, if CL(I) = INT(I), a binary signal variable, TA(I), is set to a value of 1 ("time" has run out). If this event occurs, the executive interrupts the process I at this point and control is transferred to some other part of the system, as indicated in the relevant flow charts and descriptions. The values of CL(M), CL(S) and CL(Z) are zeroed each trial, CL(A) is zeroed at the end of phase 1 (P.F.C.1.) and CL(K) at the end of phase 2 (P.F.C.1).

The executive has available a resource, RES >0, of which it receives an increment, D RES >0, for each trial.* Given initial

---

FOOTNOTE:

*For the present purpose, assume that D RES is constant. Mr. Mallen is examing an interesting system in which D RES is a function of goal proximity and, thus, by hypothesis, of motivational level.

values, the balance of RES is determined at the end of each trial by assigning a value -

RES ← RES + DRES - Σ F(I). Number of steps.

= RES + DRES - Σ F(I). CL(I) / DCL(I).

Here F(I) = Amount of effort expended in one computational step for the Ith process.

The subject parameters, SU, assign initial values to variables A(I), where -

A(I) = Amount of effort that is allocated to the Ith process and, initially, Σ A(I) = DRES.

Now effort is used to increase the rate at which a process is operated -

DCL(I) = F(I) / A(I)

For each trial there is an assignment of the form -

A(I) → A(I) + DA(I). SGN(I)

where SG(I) = +1, if TA(I) = 1, i.e. process was interrupted.

= -1, if TA(I) = 0, i.e. if not.

It will be clear that this is about the simplest control rule able to set the distribution of effort or resources as a function of demand. More refined rules, which are currently the subject of experiment, make DA(I) depend upon more intimate variables, for example, the value of DA(M) is made proportional either to the size of the working storage or to the rate at which entities decay.

5.21. Strategic Rules

At the end of each block, the automaton simulation provides last values of ρ and requires someone or something to select a subproblem for rehearsal throughout the next trial block.

In the simplest case, the subproblem class selection is performed by an external teaching machine of the sort shown in Fig.19 (this device works on the basis of a typical $C^{\wedge}$ strategy but it is even simpler to provide a teaching machine based on a Type A strategy).

Of course, such a teaching device can be "internalised", i.e. regarded as a component in the learning model which images the subject's attention directing equipment. To "internalise" the device, we provide a plan store (conceived as a part of long term memory) in which the basic plan, Type A or Type C, is lodged. This arrangement mirrors an experiment in which the subject is <u>told</u> what plan to adopt.

However, he still has to <u>use</u> the plan. The hypothesis in this matter is that he uses it as though he were a self-organising system, i.e. according to the flow chart shown in Fig.27. Hence, the Fig.27 organisation is also regarded as part of the learning model.

In practice, the estimate of H has been taken as a proficiency weighted measure. Namely:

ρ /Number of items in block with limits for advancement of $H_1 = 0.5$ and for retracking of $H_2 = 0.1$.

5.22. <u>Plan Building</u>

The learning model can also construct a plan of its own and place it in the plan store. This activity has not yet been examined in detail but the broad principles are clear and simple. Notice first that neither the real subject nor the model is called upon to create a plan from nothing. A description of the task is <u>given</u> (as in Section 1.4 and 1.5). In the learning model, this is part of the representation scheme of 5.6 which is assumed to reside in long term memory and which is freely used in problem decomposition (the whole part <u>relationships</u> inherent in one view of the task are provided). Next, it should be noted that the model is only required to construct a plan after it has had some preliminary experience in dealing with the problems posed by the task. Hence, at the moment it is asked to produce a plan for further learning, the model will already have used its decomposition rules in order to build up operators and strings. Finally, any plan is a generalisation of some subset of the decomposition rules applicable to 4-lamp problems.

To see this, notice that decomposition rules are applied to individual (4-lamp) problems and reduce them to problems of a lower order. Thus, if $x \varepsilon X_{ABCD}$,

$$x \to x_1, x_2, \quad x_1 \varepsilon X_{ABC}, \quad x_2 \varepsilon X_D$$
$$x_1 \to x_3, x_4, \quad x_3 \varepsilon X_{AB}, \quad x_4 \varepsilon X_C$$
$$x_3 \to x_5, x_6, \quad x_5 \varepsilon X_A, \quad x_6 \varepsilon X_B \quad \ldots\ldots\ldots (I)$$

is one set of rewriting rules, which reduces the problem to a collection of order 1 problems, and,

$$x \to x_7, x_8, \quad x_7 \varepsilon X_{CD}, \quad x_8 \varepsilon X_{AD}$$
$$x_7 \to x_9, x_{10}, \quad x_9 \varepsilon X_C, \quad x_{10} \varepsilon X_D$$
$$x_8 \to x_{11}, x_{12}, \quad x_{11} \varepsilon X_A, \quad x_{12} \varepsilon X_B \quad \ldots\ldots\ldots (II)$$

is another set with the same property.

Now, clearly, if the entire class of (4-lamp) problems is decomposed according to (I) or (II), (i.e. the same decomposition procedure is employed for all $x \ X_{ABCD}$) then it is legitimate to write the generalised decomposition in terms of the attributes describing the class itself:

$$ABCD \longrightarrow ABC, D$$
$$ABC \longrightarrow AB, C$$
$$AB \longrightarrow A, B \qquad \ldots\ldots\ldots (I^*)$$

or:

$$ABCD \longrightarrow AB, CD$$
$$CD \longrightarrow C, D$$
$$AB \longrightarrow A, B \qquad \ldots\ldots\ldots (II^*)$$

Further, reversing the order of the generalised decomposition and eliminating repetitions, we obtain a Type A plan from (I^*) and a Type C plan from (II^*). Thus:

A, B, AB, C, ABC, D, ABCD

and

A, B, AB, C, D, CD, ABCD

By hypothesis, these are the plans that would be conceived by subjects who adopt a uniform method of decomposition for all the problems. Subjects who do not do so (and, in particular, subjects who fail to respect the complementarity of the subsets) are likely to produce plans of Type B. By way of a mechanism, it is proposed that the subject inspects the previously used 4-lamp decomposition rules and selects the most frequently used (in a pure case, the unique) category to which they belong. This is the generalised decomposition and (read backwards) this is the plan.

Similar considerations apply to an inspection of the operators and operator strings available in working storage, (here, of course, the utilisations provide the frequency count automatically). A simple generalisation from the repertoire yields a plan.

In either case(generalisation from used decomposition rules, or from the repertoire of operators and operator strings) plan construction is readily carried out and amounts to no more than an application of the analogy operation to the domain of working storage giving an output to be lodged in the plan store.

5.23. Experimental Results

The output from the learning model consists in a sequence of trial data (as already detailed in the case of a real life subject) together with an optional print out of the contents of working storage (the names, domains, ranges, lifespans and utilisations of all L^0 operators and L^0 operator strings). It is also possible to obtain a record of the path taken in the L^1 processing.

The main predictions derived from the simulation experiments were presented in Section 2. The following notes outline the type of work which has so far gone on.

1. Decay: With low decay (or ready access to reproduction) the model learns rapidly and is not embarrassed by difficult problems. Consequently, it spends a disproportionately large percentage of trials dealing with simple (especially single attribute) problems. Its behaviour is thus thoroughly unrealistic, either with respect to a Type A or a Type C strategy. But the lack of realism is especially marked in the case of a Type C strategy when the model's curve, expressed in the format of Appendix 3 or Appendix 4, is convex rather than concave. Moreover, in these conditions, the model does not need to alternate AB and CD problems, since there is substantially no interference. Inspection of the working store printout reveals the presence of many (typically 12 - 15) operators and many (typically 7 or 8) strings. These entities tend to persist.
   As the decay is increased, the model's behaviour does become realistic. It is embarrassed by difficult problems and interference is a real handicap. Eventually, if decay is still further increased, the model behaves unrealistically for a different reason. It cannot hold enough entities in its working store (typically 4 operators and no strings or only transient strings).

2. Strategies: Both Type A and Type C strategies have been used in the experiments. With suitably chosen decay (or reproduction) values, these produce grossly realistic behaviours. The relative merits of these strategies depend chiefly upon the substitution threshold and the extension parameter.

(continued)

3. Parameters: If the model is biassed to extend strings, it is able to learn most efficiently with a Type A strategy. Conversely, if it is biassed to substitute strings and build up complex operators, it is at an aivantage with a Type C strategy. These differences are just detectable in low decay conditions (for example, inhibiting substitution, delays a Type C learning process) but they become clear as decay is increased to a realistic level (in the sense of (1) above).

4. Knowledge of Results: If the cost of knowledge of results processing is low, then the model learns more rapidly, when detailed knowledge of results is provided and the working store contents show that this information is used to construct competent operators or strings. As the cost of knowledge of results is increased, the detailed information is used only when the model runs into trouble and, consequently, the provision of detailed knowledge of results has less influence upon the number of trials required to reach criterial performance. In view of the data processing required (PFC 7(2)), it is reasonable to hypothesise that any use of detailed knowledge of results is a lengthy and costly process.

The printout from the simulation experiments is too bulky to reproduce in this report (since in order to be more meaningful than a summary statement, it needs to be exhibited in some detail). However, the current results, together with the TELCOMP version of the programme, are lodged in Washington and are available at this laboratory. The interested reader is also referred to work with earlier learning models simulating the skills described in Ref.1 and Ref.2. This material is most readily accessible in the Final Scientific Report of Contract AF-61-052-640, August, 1966.

5.24. Main Deficiencies

The model offers a reasonably veridical picture of the learning that goes on in a rule application task and it provides useful, if not essential, data for the teaching system designer. But, in principle, it has far greater generality as a model for concept learning and skill learning. Viewed in this light, it has, however, several outstanding deficiencies. Most of them can easily be remedied and the work necessary to do so is in progress.

(1) Although certain information is retained in immediate memory, insufficient attention is paid (in the model) to the limitations of immediate memory. At various places in the system, symbols should be retained in finite sized lists (or stacks) rather than indefinitely sized lists. This is especially true of the representation of problem decomposition.

(2) It is very easy to make a formal gesture of recognition over the issue of long term memory. Thus, it is quite a trivial matter to place an entity, that has appeared in the awaiting reproduction stack on M or more occasions, in a special list (not subject to decay) called the long term memory list.

This expedient can be quite useful (as in (5) below) but is not really a fair response to the requirement for a long term memory. To make a fair response, it is necessary to incorporate an abstractive mechanism into the model and I currently conceive something like Hunt, Main and Stones' CLS programme abstracting from the repertoire of operators and strings in working storage (abstracting from intermediate memory).

(3) The analogy operation is not yet fully exploited (it was not even described in the present account though it has been used in rough simulations of double rule learning to effect positive transfer of training between the subskills).

(4) As mooted in 5.21., plan construction is a perfectly straightforward process. The trouble is that the simple expedient of constructing a plan analagous to the most often used set of decompositions, is probably inadequate. Some criterion other than frequency of use of a decomposition and (to judge by our subjects) other than a simple minimal "success" of the decomposition, is required and further work is needed in this direction.

(5) The most serious criticism of the learning model is its failure to internalise goals. Fortunately, it is designed in such a way that a goal internalisation mechanism can be added; the mechanism is outlined in the next section.

5.25. Goal Internalisation

The model "knows" a skill when it has the operator strings needed to solve the problems posed in the conduct of the skill. It may, of course, be certain but mistaken (in which case, providing it attends to knowledge of results, its mistake will be corrected by the knowledge of results signal). Again, it may be uncertain (in the sense that guessing is involved). If so, the knowledge of results signal serves to resolve its uncertainty. But there is no mechanism that leads it to believe it is right (or not). In a sense, the model always believes it is right unless it is in a state of uncertainty (the oddity of this failure to entertain beliefs is particularly evident, if we consider the case when knowledge of results is not provided).

Real subjects do entertain beliefs about the rectitude of their responses, i.e. they internalise goals and derive internal corrective signals from goal comparisons. Further, the internalised goal guides their behaviour in the absence of a knowledge of results signal. Often it happens that the internal corrective

signal is delivered after the event. It is as though the subject responded rapidly and later checked his response against a slower process entailing some deliberation.

Now the learning model may have, and, in general, does have, more than one way of solving a problem. In general, it is also true that it has a slow and primitive problem solving method (such as the application of a sequence of order 1 operators, so slow that it is useless in the conditions of the experiment) together with a sophisticated and fast process (for example, the application of complex operators or operator strings).

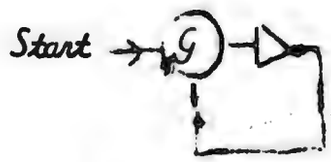
My hypothesis is simply this. The slow and sure process becomes lodged in long term memory. It is applied in parallel with the sophisticated fast process. Later in learning, the output of the slow process is compared with the output of the fast process and, if the outputs agree, then an internal goal is achieved, i.e. the subject expects the knowledge of results signal to confirm his rectitude.

Conversely, in the absence of a knowledge of results signal, the existence of the slow process, embodied in long term memory, allows for continued performance and the reconstruction of the skill at a later date.

5.26 Broad Picture

Although the learning model has been presented as a simulation of learning in a rule application task, it is intended to have a much wider field of application. Given the modifications of 5.23 and 5.24, it should prove competent in cases where problems do not have unique solutions. Moreover, it reflects processes that are ubiquitous in skill and concept learning. By token of this, the string construction routines can be taken as paradigms for sequential and mnemonic learning, whilst the higher level operator routines, obtained by substitution, are paradigms for wholistic visual or spatial learning.

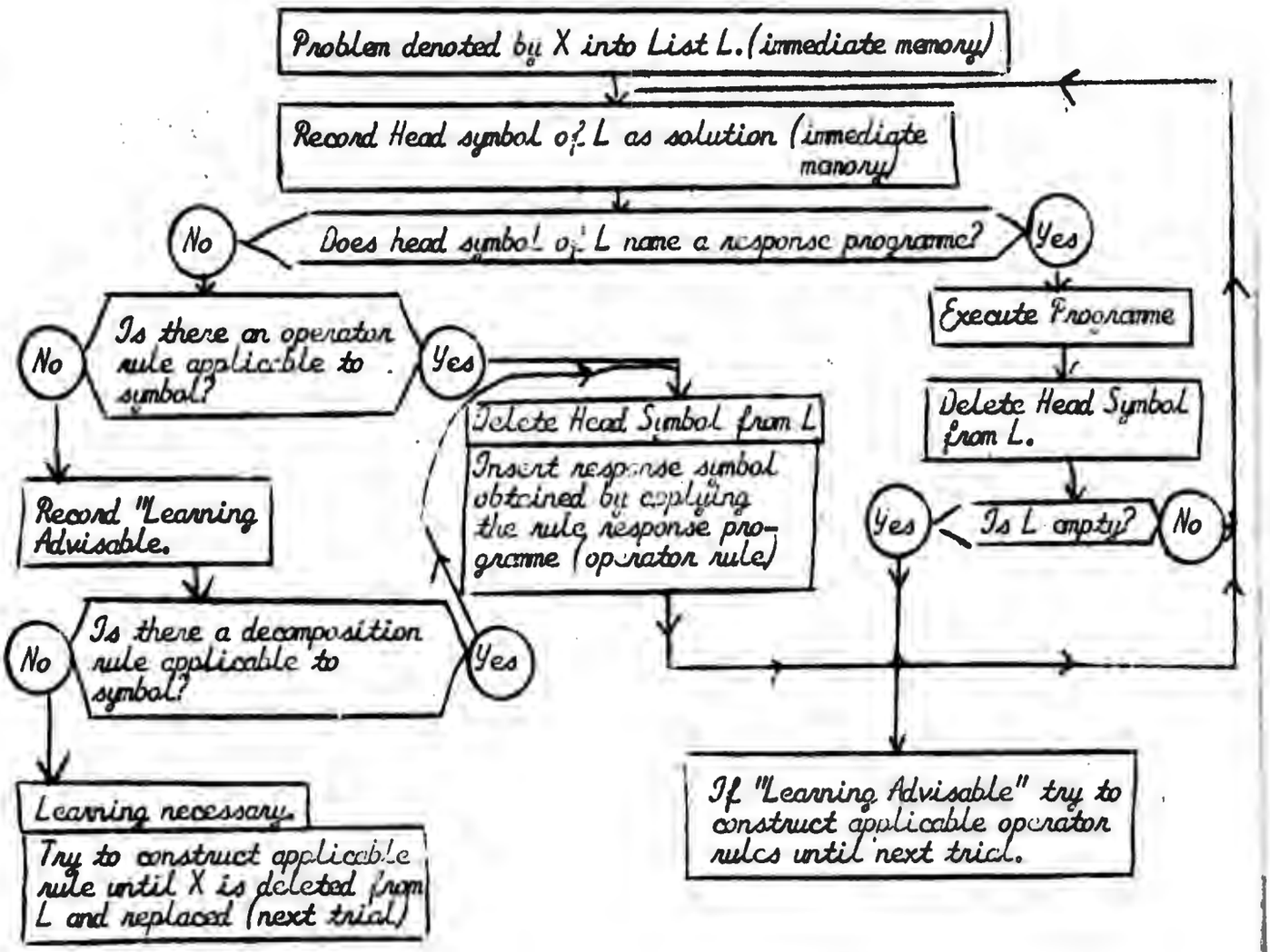
Figure 28



G = Goal of unit.

Figure 28. A TOTE Unit.

Figure 29.



Some Typical Decomposition Rules: $[ABCD] \rightarrow [AB], [CD]$ $[ABC] \rightarrow [AB], [C]$
$[ABCD] \rightarrow [ABC], [D]$ $[AB] \rightarrow [A], [B]$

Some Typical Operator Rules $[AB] \rightarrow [\alpha, \beta]$ $[B] \rightarrow [\beta]$ $[C] \rightarrow [\gamma]$
$[A] \rightarrow [\alpha]$ $[D] \rightarrow [\delta]$

$\alpha, \beta, \gamma, \delta$ stand for certain response programmes.

Figure 29. A Simple Push Down List Automaton for Interpretative Process.

Figure 30.

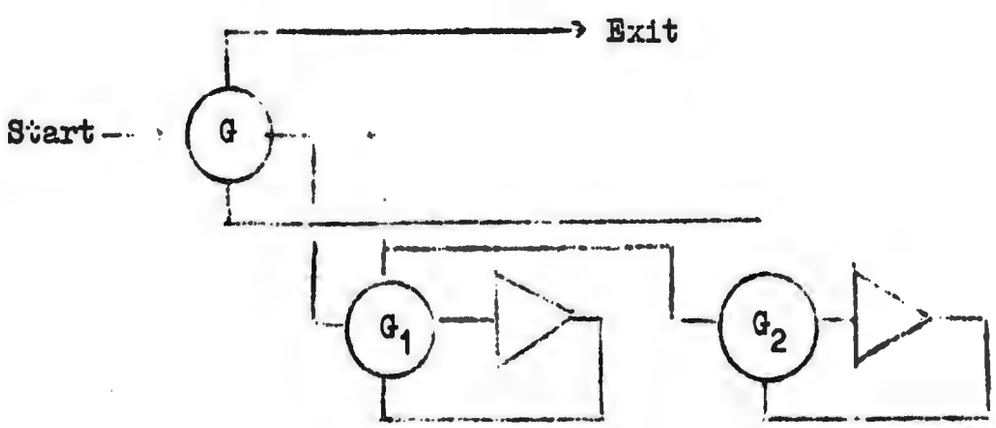


Figure 30. An Hierarchy or Plan of Tote Units.

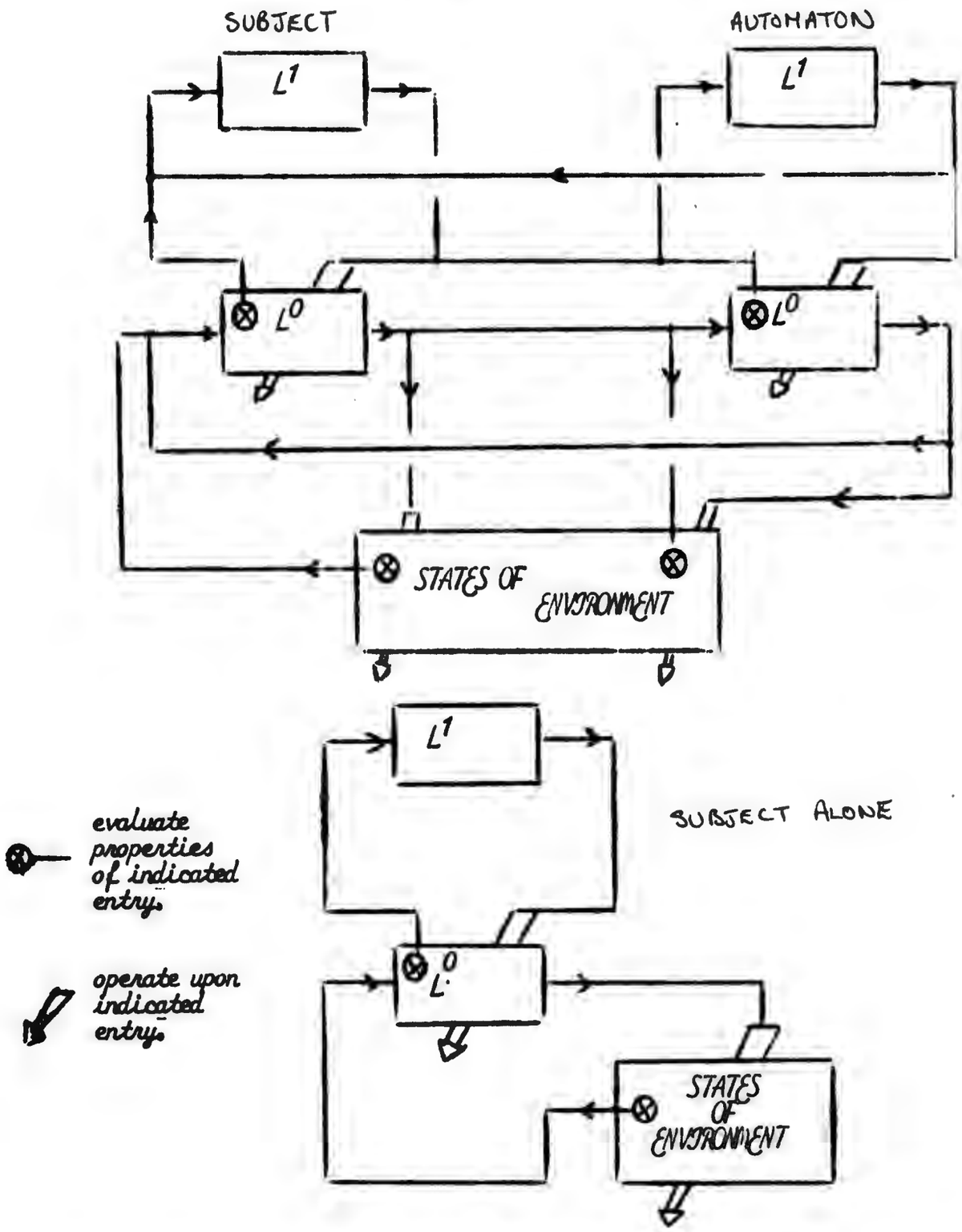


Fig. 31. The Type of Control Hierarchy Supposed to Exist in the Subject.

FIGURE 32.

Stimulus Numbers, SN.

	X_1				X_2						X_3				X_4
SN.	X_A	X_B	X_C	X_D	X_{AB}	X_{AC}	X_{AD}	X_{BC}	X_{BD}	X_{CD}	X_{ABC}	X_{ABD}	X_{ACD}	X_{BCD}	X_{ABCD}
1	1	3	5	7	9	13	17	21	25	29	33	41	49	57	65
2	2	4	6	8	10	14	18	22	26	30	34	42	50	58	66
					11	15	19	23	27	31	35	43	51	59	67
					12	16	20	24	28	32	36	44	52	60	68
											37	45	53	61	69
											38	46	54	62	70
											39	47	55	63	71
											40	48	56	64	72

Refer to Fig.7.

List of Response Programme Names. An open ended list with initial entries of names of response programmes producing 8 responses, y = 1,0,0,0,0,0,0,0.....
....y = 0,0,0,0,0,0,0,1.

Subproblem Class Number: i = A,B,.....AB, AC,.....ABC,ABD....ABCD.

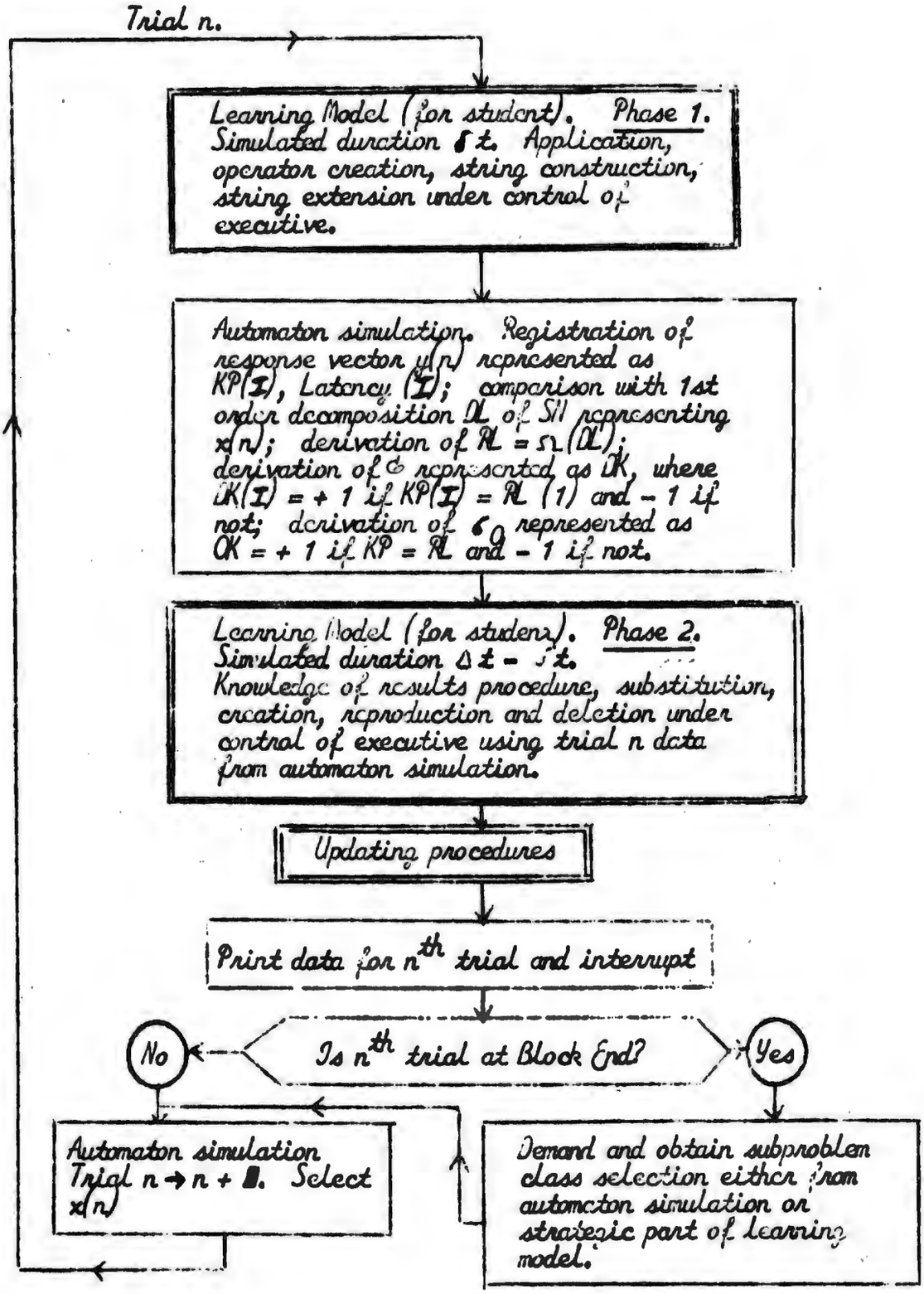
Ω specified by initial l^0 operators.

Subject parameters SU. A vector of 16 variable assigned initial values but variable whilst programme is executed

σ , Ω (x), ρ , Last Value ρ, represented (as in the text) by variables "OK", "DKR" etc.

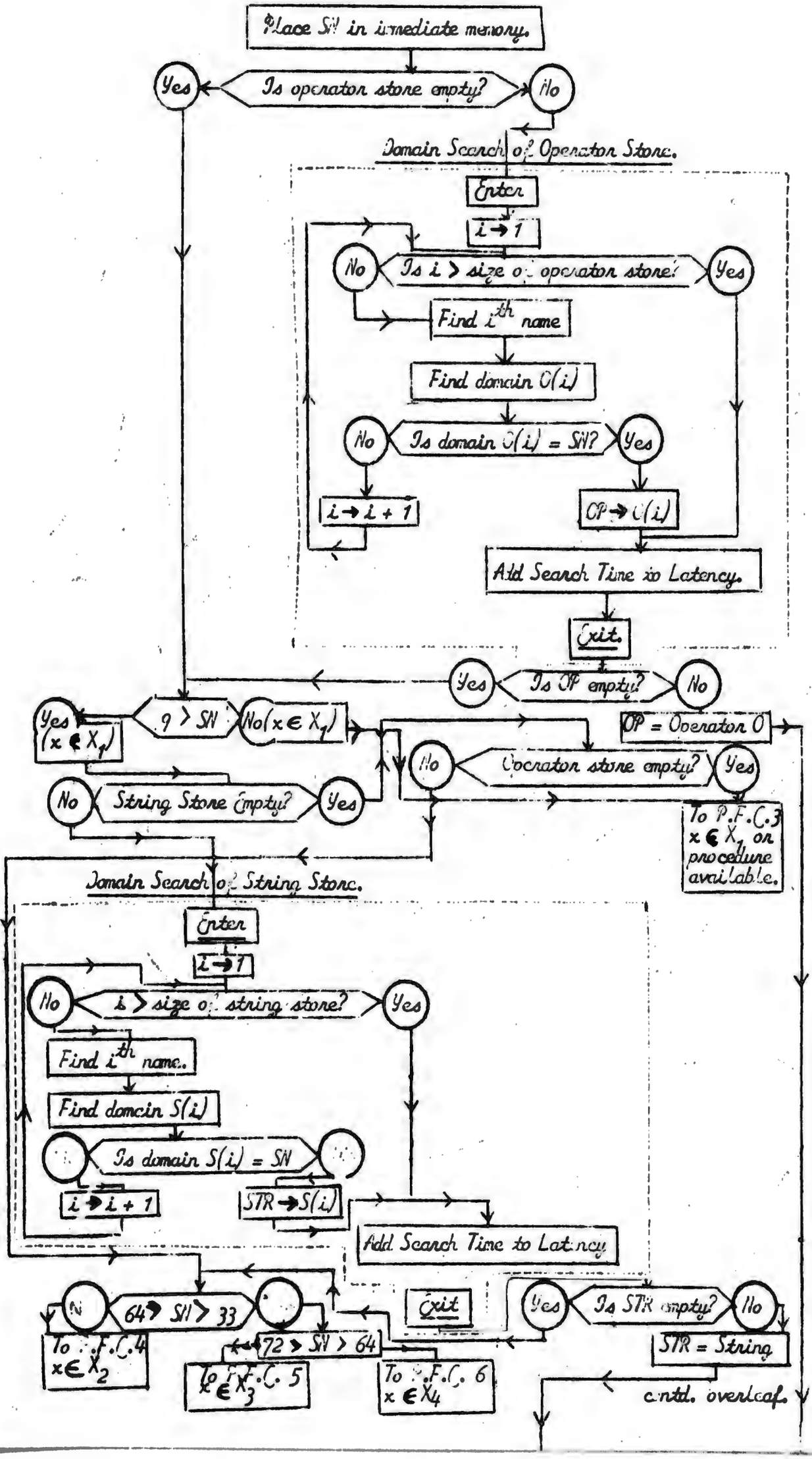
Figure 32. Representation of Variables in Simulation.

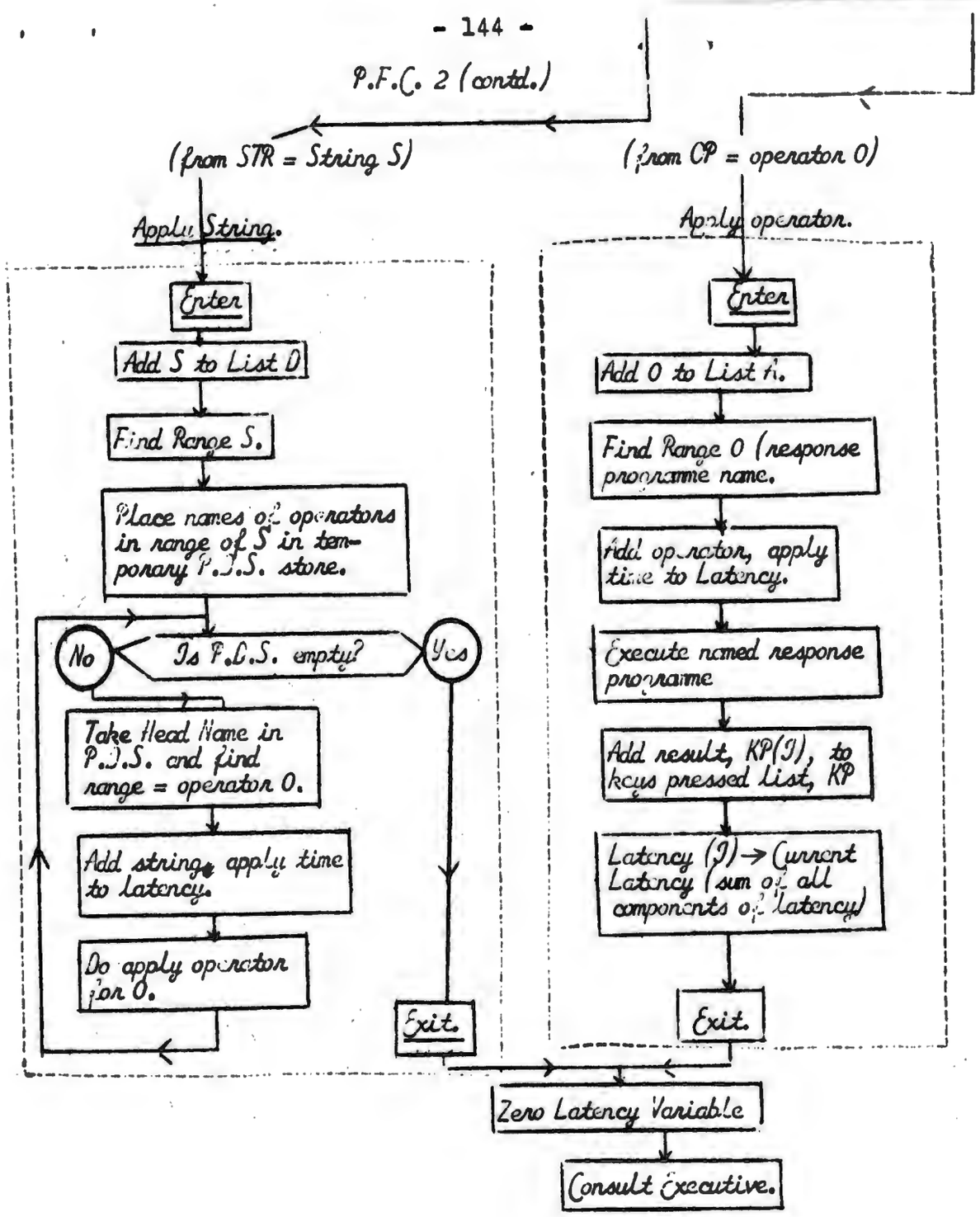
P.F.C. 1.



P.F.C. 1. Overall Organisation.

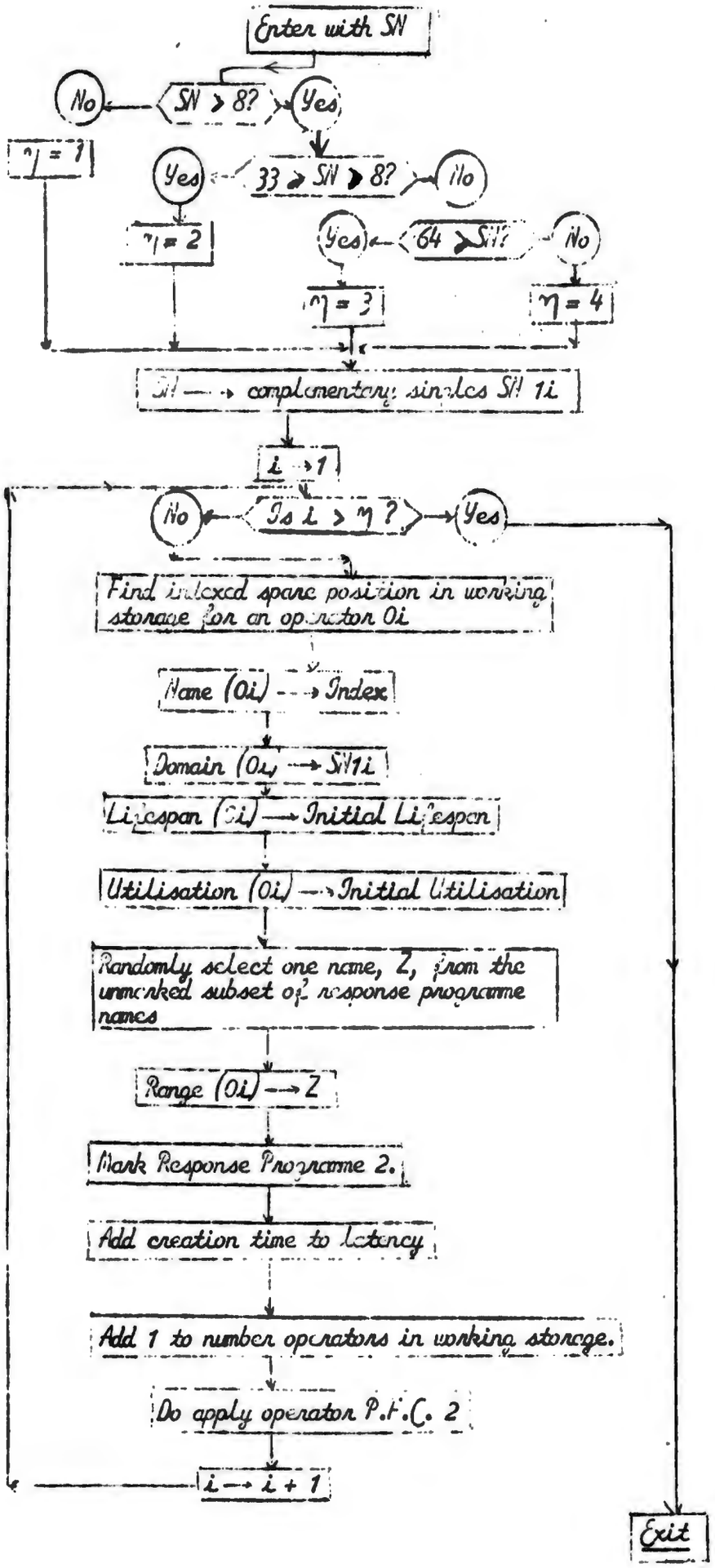
P.F.C. 2.

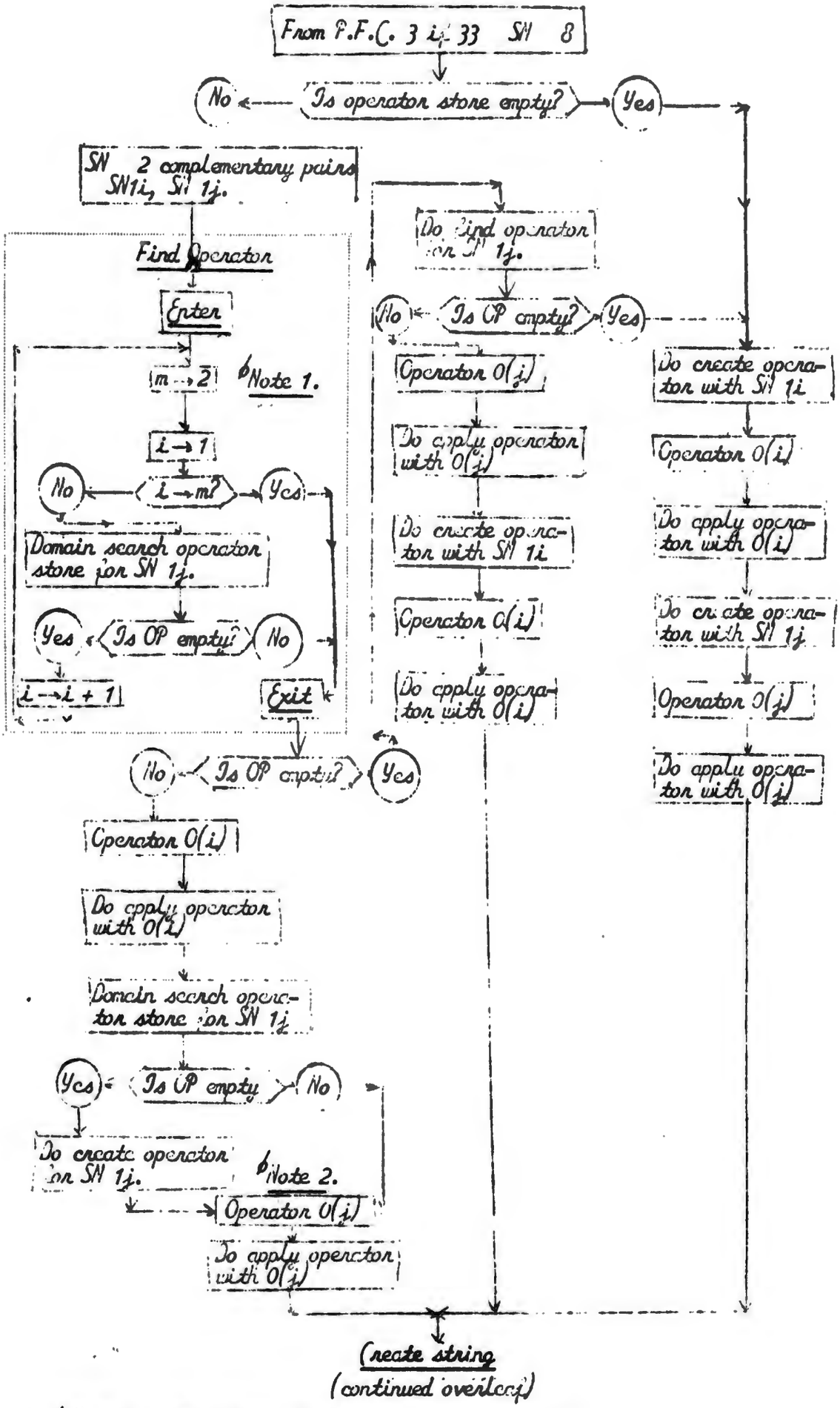

Place SN in immediate memory.
Is operator store empty?
Yes
No
Domain Search of Operator Store.
Enter
i → 1
Is i > size of operator store?
No
Yes
Find i^th name
Find domain O(i)
Is domain O(i) = SN?
No
Yes
i → i + 1
OP → O(i)
Add Search Time to Latency.
Exit.
Is OP empty?
Yes
No
OP = Operator O
Operator store empty?
Yes
No
To P.F.C.3
x ∈ X1 or procedure available.
q > SN
Yes (x ∈ X1)
No (x ∈ X1)
String Store Empty?
Yes
No
Domain Search of String Store.
Enter
i → 1
i > size of string store?
No
Yes
Find i^th name.
Find domain S(i)
Is domain S(i) = SN
i → i + 1
STR → S(i)
Add Search Time to Latency
Exit
64 > SN > 33
72 > SN > 64
To P.F.C.4 x ∈ X2
To P.F.C. 5 x ∈ X3
To P.F.C. 6 x ∈ X4
Is STR empty?
Yes
No
STR = String
cntd. overleaf.



P.F.C. 2. Application Procedure.

P.F.C. 3.

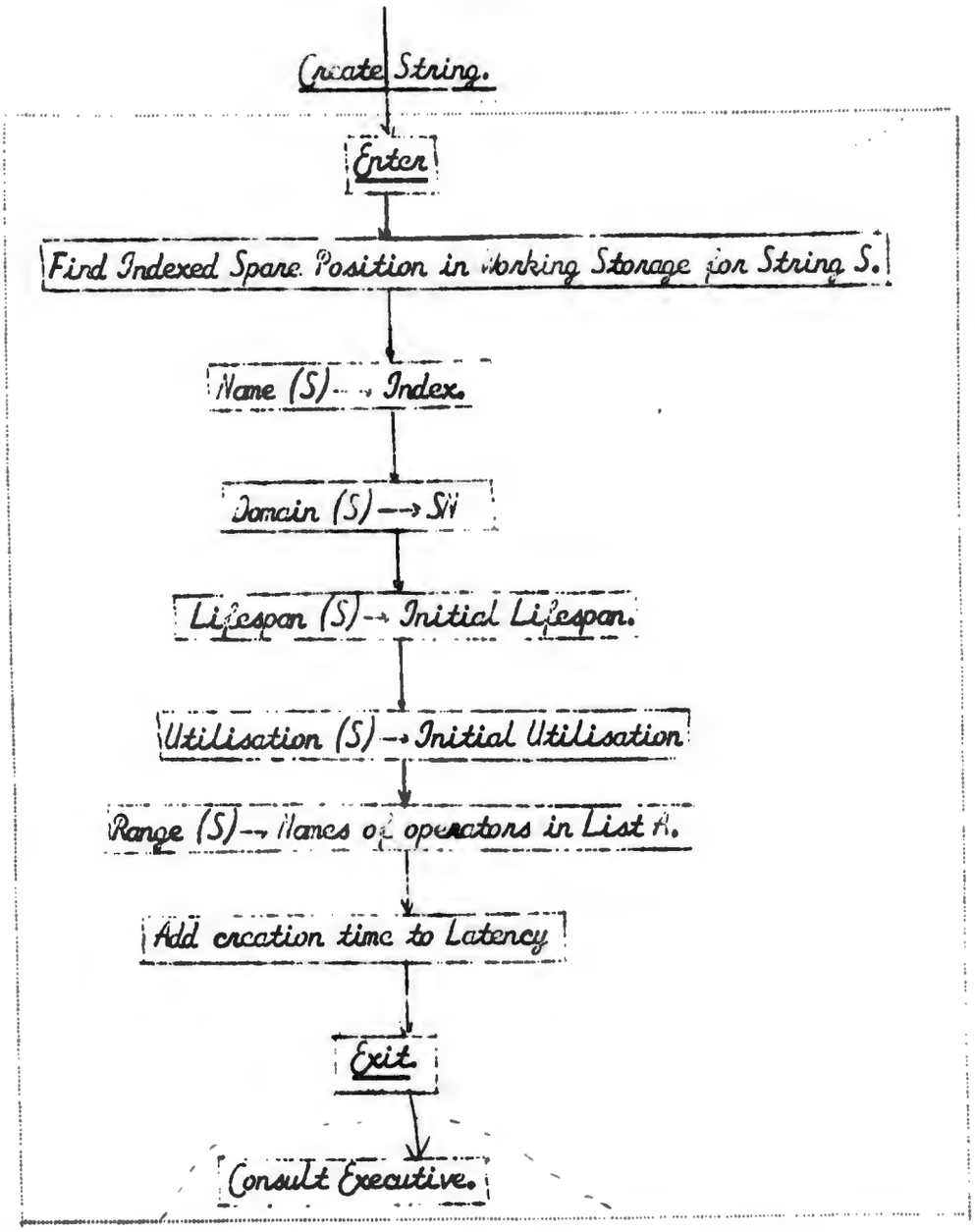

Enter with SN
SN > 8?
No
Yes
η = 1
33 > SN > 8?
Yes
No
η = 2
64 > SN?
Yes
No
η = 3
η = 4
SN → complementary singles SN 1i
i → 1
Is i > η ?
No
Yes
Find indexed spare position in working storage for an operator Oi
Name (Oi) → Index
Domain (Oi) → SN1i
Lifespan (Oi) → Initial Lifespan
Utilisation (Oi) → Initial Utilisation
Randomly select one name, Z, from the unmarked subset of response programme names
Range (Oi) → Z
Mark Response Programme Z.
Add creation time to latency
Add 1 to number operators in working storage.
Do apply operator P.F.C. 2
i → i + 1
Exit



Note 1. m = number of elements in decomposition.

Note 2. Originally i → 1 before creation. In later version, one value of i is selected randomly. The random selection before creation makes a difference when the model only has a chance to create one operator, i.e. when the executive interrupts it.

P.F.C. 4 (continued)

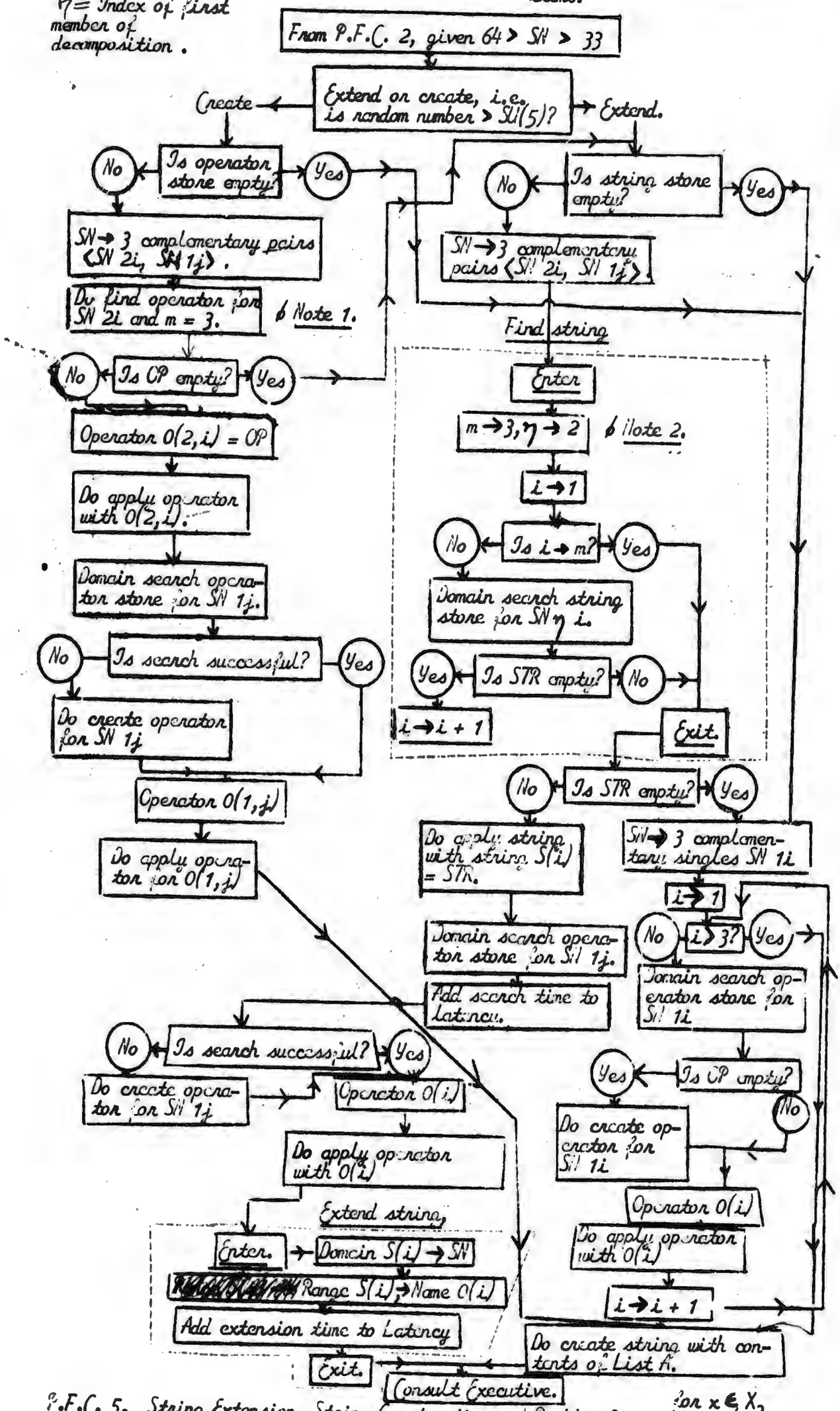


P.F.C. 4. String Construction and Problem Decomposition for $x(n) \in X_2$

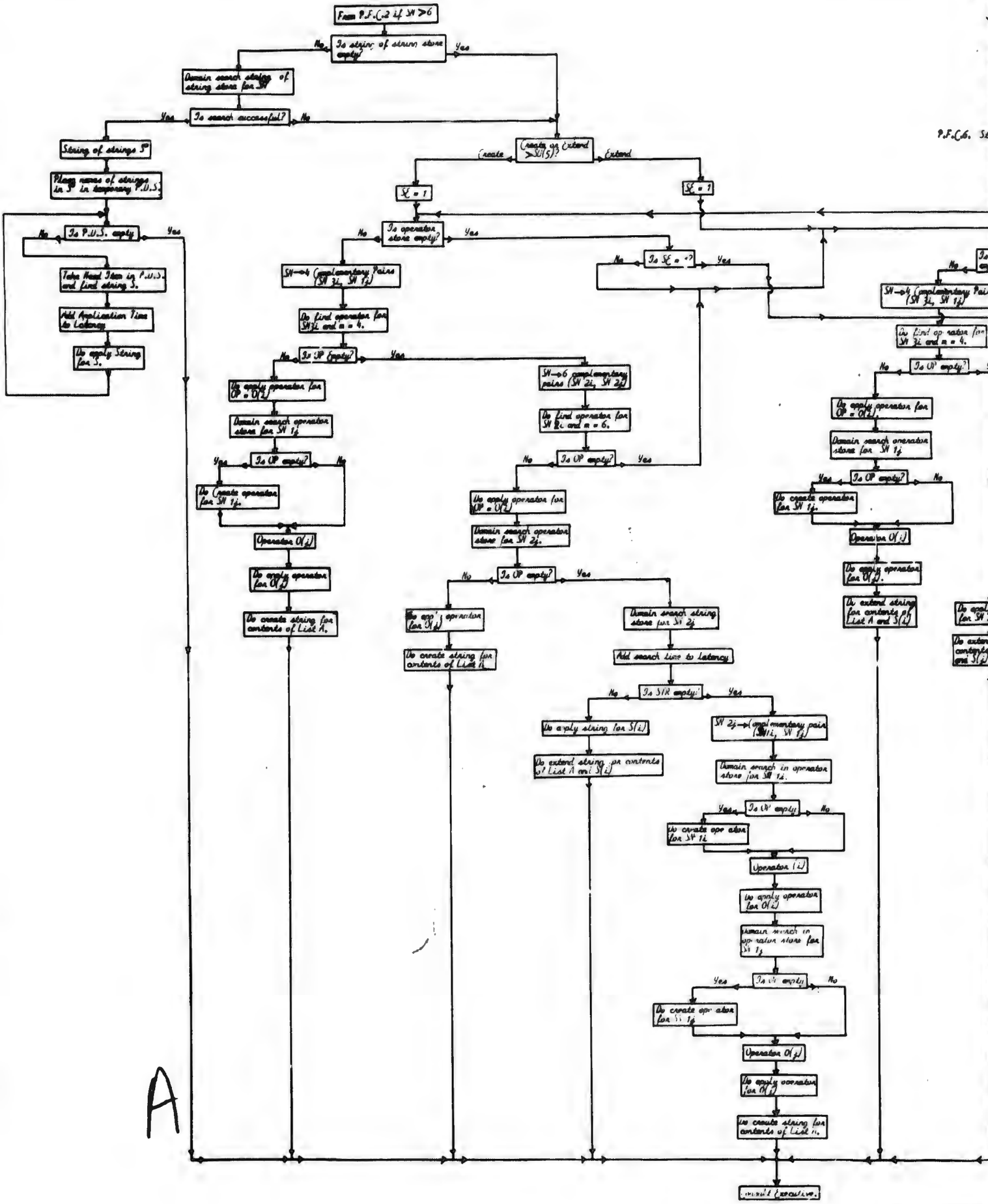
P.F.C. 5.

Note 1.
m = number of complementary items.

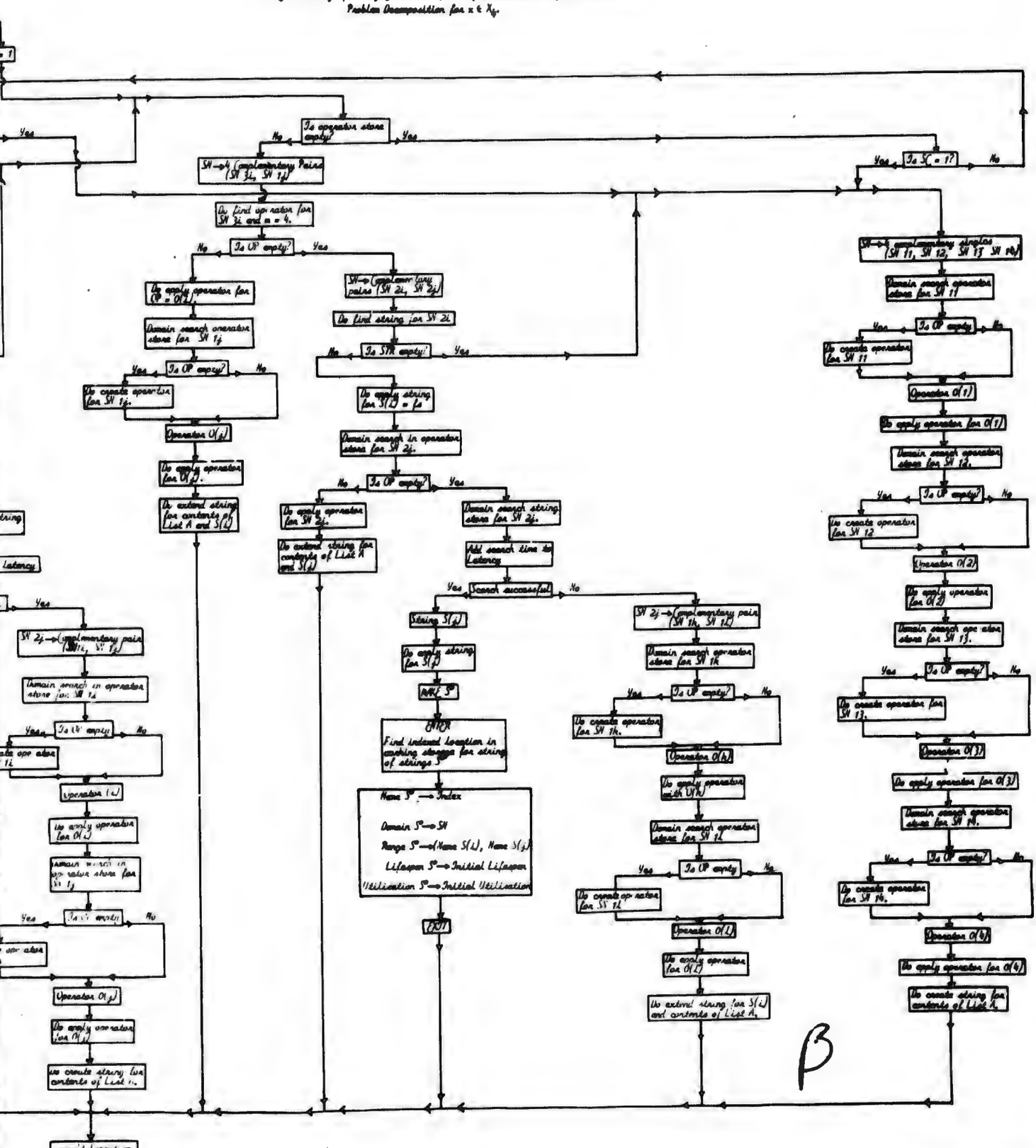
Note 2.
m = number of complementary items.
η = Index of first member of decomposition.



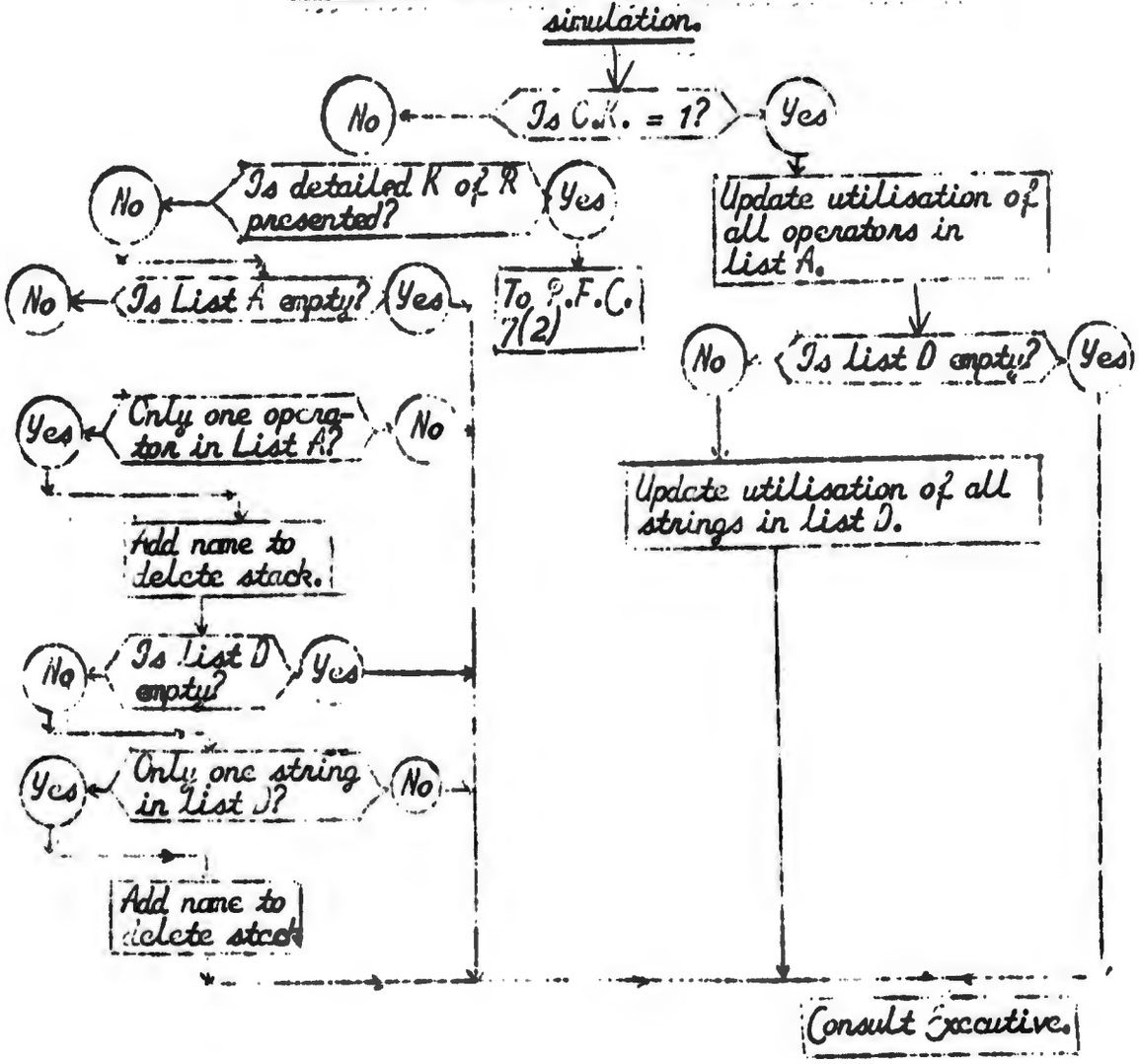
P.F.C. 5. String Extension, String Construction and Problem Decomposition for $x \in X_3$


From P.F.C.2 if SW > 6
Is string of strings store empty?
No
Yes
Domain search strings of string store for SW
Is search successful?
No
Yes
String of strings S'
Place names of strings in S' in temporary P.U.S.
Is P.U.S. empty
No
Yes
Take Head Item in P.U.S. and find string S.
Add Application Time to Latency
Do apply String for S.
(Create or Extend >SC(5)?
Create
Extend
SC = 1
SC = 1
Is operator store empty?
No
Yes
SW → 4 complementary Pairs (SW 2i, SW 1j)
Do find operator for SW1i and n = 4.
Is OP empty?
No
Yes
Do apply operator for OP = O(j)
Domain search operator store for SW 1j
Is OP empty?
Yes
No
Do Create operator for SW 1j.
Operator O(j)
Do apply operator for O(j)
Do create string for contents of List A.
SW → 6 complementary pairs (SW 2i, SW 2j)
Do find operator for SW 2i and n = 6.
Is OP empty?
No
Yes
Do apply operator for OP = O(j)
Domain search operator store for SW 2j.
Is OP empty?
No
Yes
Do app' operator for SW j
Do create string for contents of List A
Domain search string store for SW 2j
Add search time to Latency
Is S/R empty?
No
Yes
Do apply string for S(i)
Do extend string for contents of List A and S(i)
SW 2j → (complementary pairs (SW1i, SW 1j)
Domain search in operator store for SW 1i.
Is OP empty
No
Yes
Do create operator for SW 1i
Operator (i)
Do apply operator for O(i)
Domain search in operation store for SW 1j
Is OP empty
No
Yes
Do create operator for SW 1j
Operator O(j)
Do apply operator for O(j)
Do create string for contents of List A.
SW → 4 complementary Pairs (SW 2i, SW 1j)
Do find operator for SW 2i and n = 4.
Is OP empty?
No
Do apply operator for OP = O(j)
Domain search operator store for SW 1j
Is OP empty?
Yes
No
Do create operator for SW 1j.
Operator O(j)
Do apply operator for O(j).
Do extend string for contents of List A and S(j)
A

P.F.C6. String of String Construction, String Extension, String Construction and Problem Decomposition for $x \in X_6$.

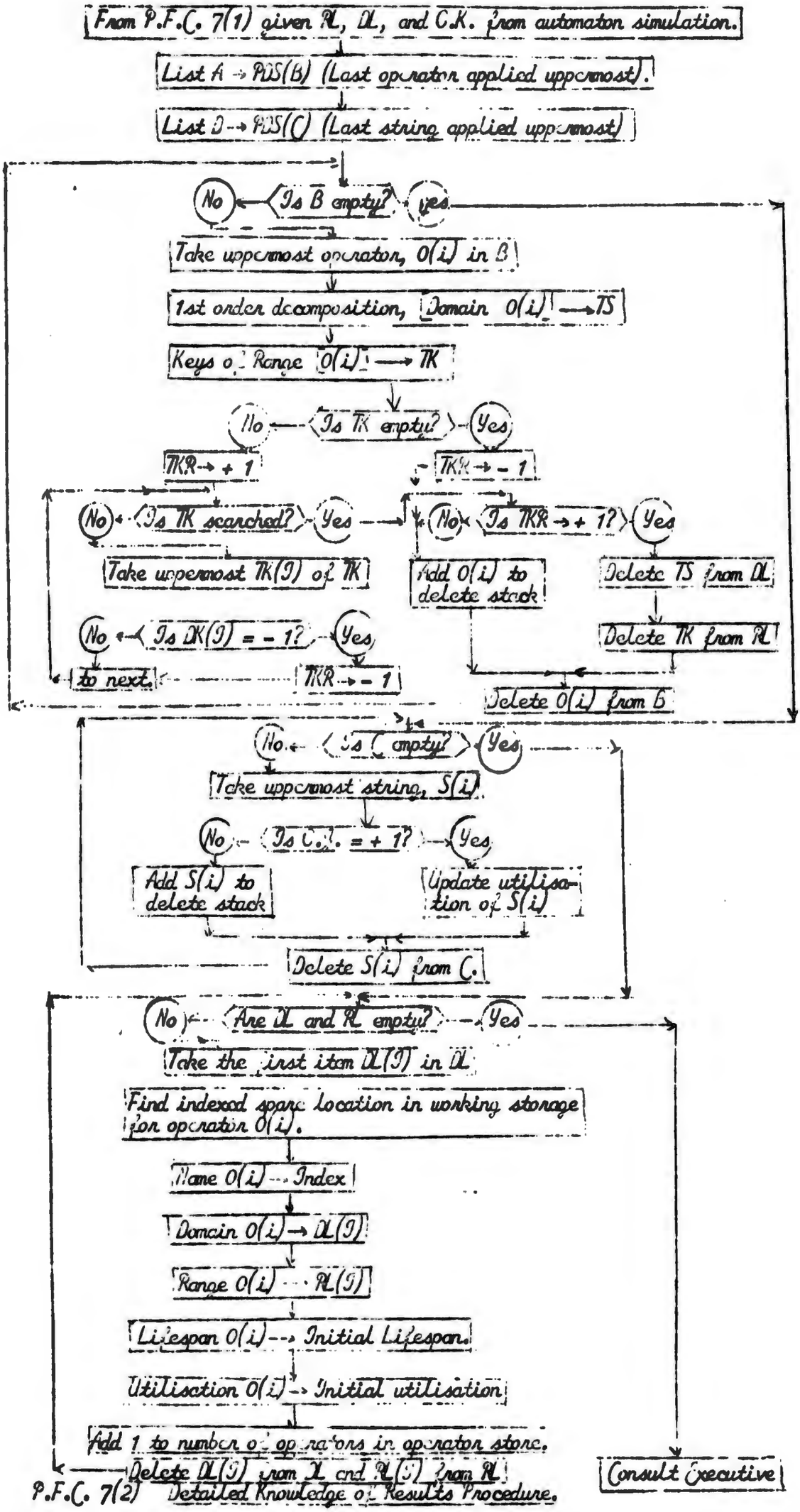


P.F.C. 7(1)

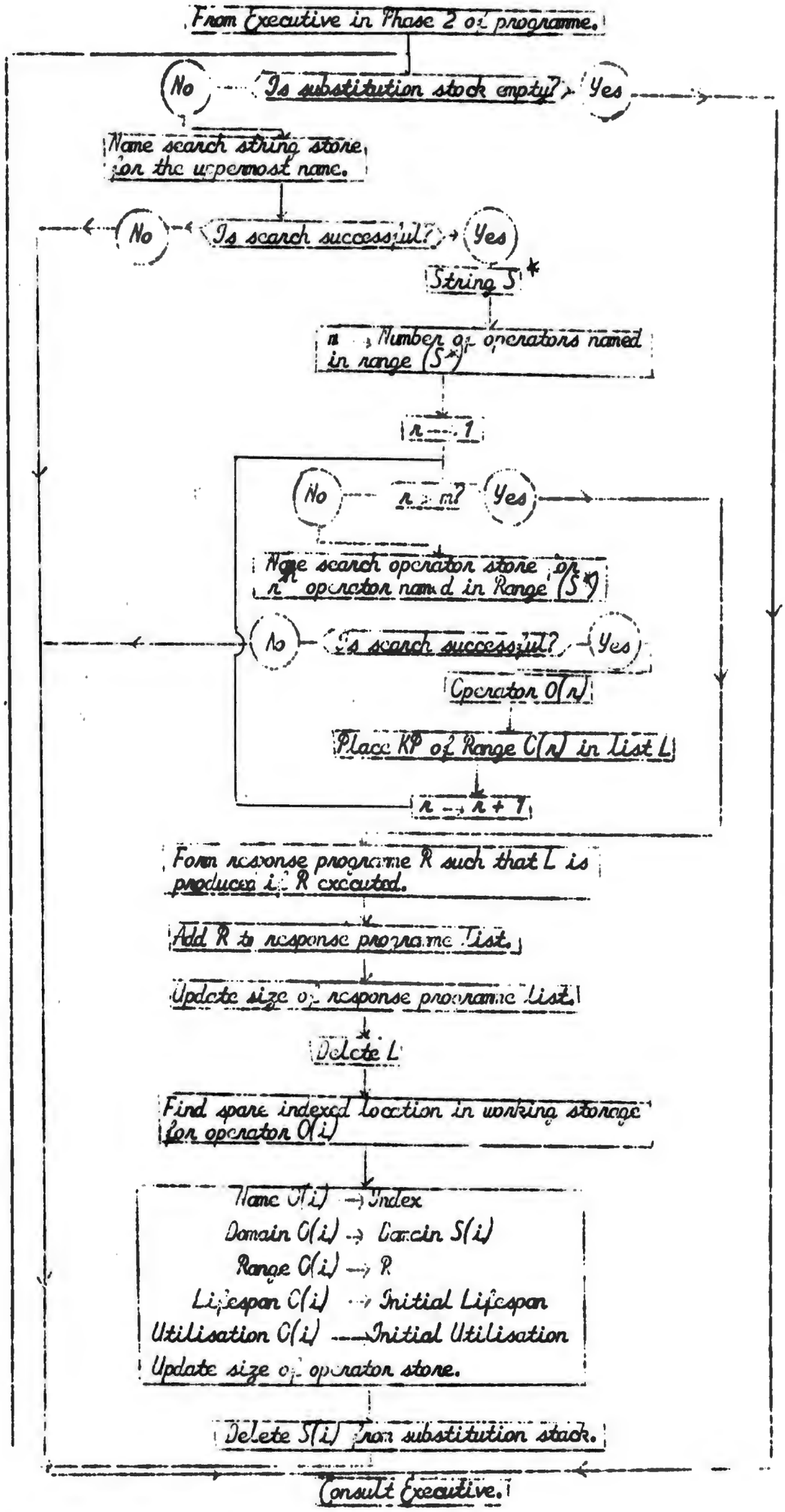


P.F.C. 7(1) Complete knowledge of Results Processing.

P.F.C. 7(2)

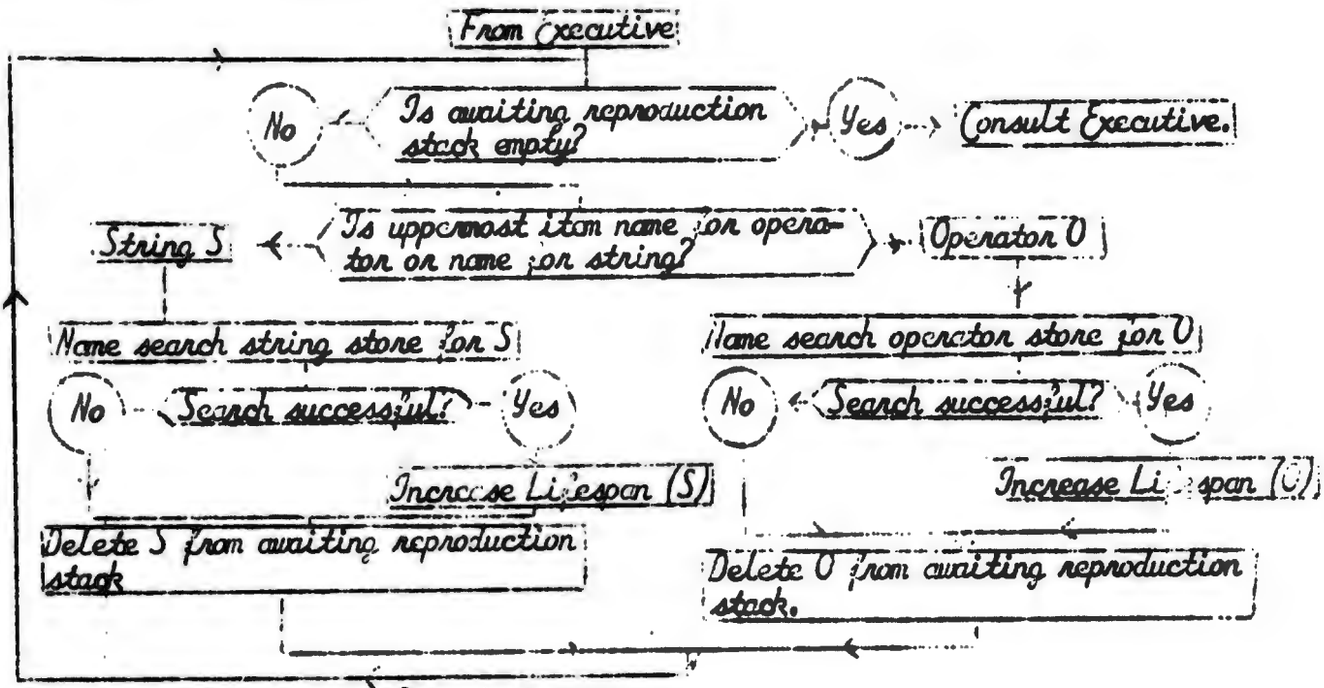

From P.F.C. 7(1) given RL, DL, and C.K. from automaton simulation.
List A → PDS(B) (Last operator applied uppermost).
List D → PDS(C) (Last string applied uppermost)
No
Is B empty?
yes
Take uppermost operator, O(i) in B
1st order decomposition, Domain O(i) → TS
Keys of Range O(i) → TK
No
Is TK empty?
Yes
TKR → + 1
TKR → - 1
No
Is TK searched?
Yes
No
Is TKR → + 1?
Yes
Take uppermost TK(I) of TK
Add O(i) to delete stack
Delete TS from DL
No
Is DK(I) = - 1?
Yes
Delete TK from RL
to next
TKR → - 1
Delete O(i) from B
No
Is C empty?
Yes
Take uppermost string, S(i)
No
Is C.I. = + 1?
Yes
Add S(i) to delete stack
Update utilisation of S(i)
Delete S(i) from C.
No
Are DL and RL empty?
Yes
Take the first item DL(I) in DL
Find indexed spare location in working storage for operator O(i).
Name O(i) → Index
Domain O(i) → DL(I)
Range O(i) → RL(I)
Lifespan O(i) → Initial Lifespan.
Utilisation O(i) → Initial utilisation
Add 1 to number of operators in operator store.
Delete DL(I) from DL and RL(I) from RL
Consult Executive
P.F.C. 7(2) Detailed Knowledge of Results Procedure.

P.F.C. 8.



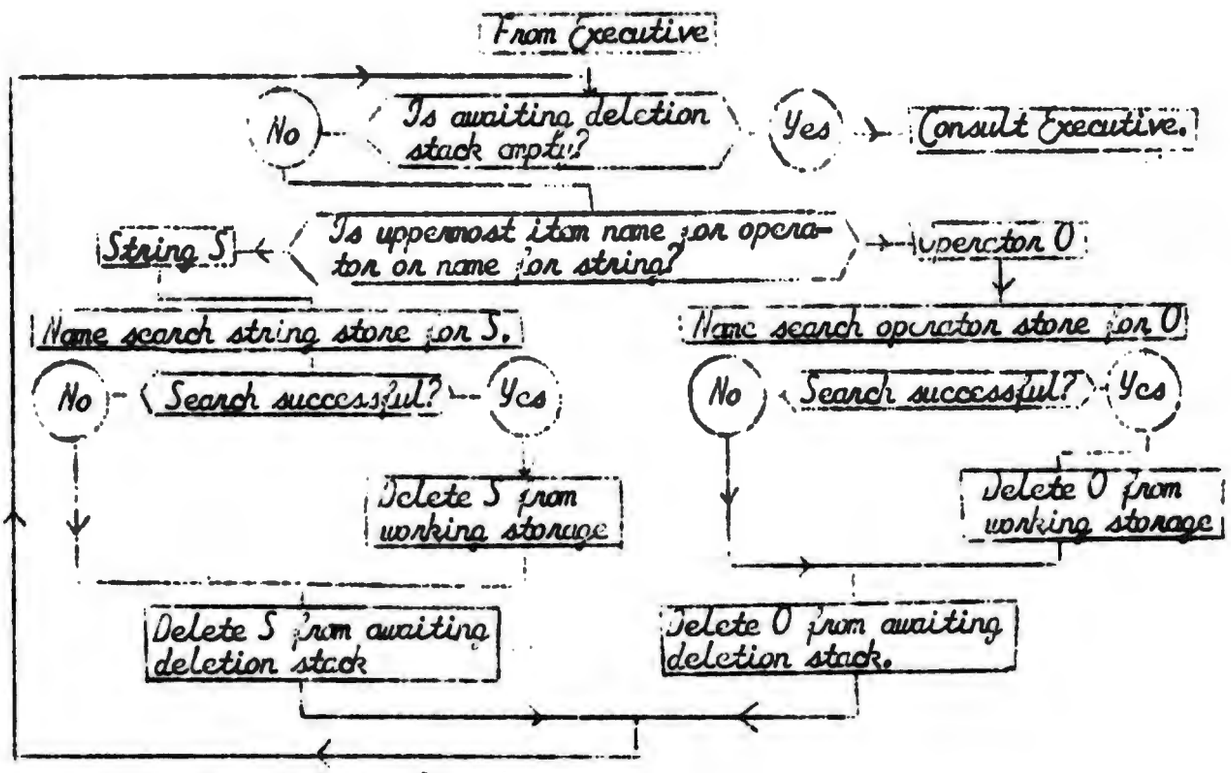
P.F.C. 8. Substitution Procedure.

P.F.C. 9.



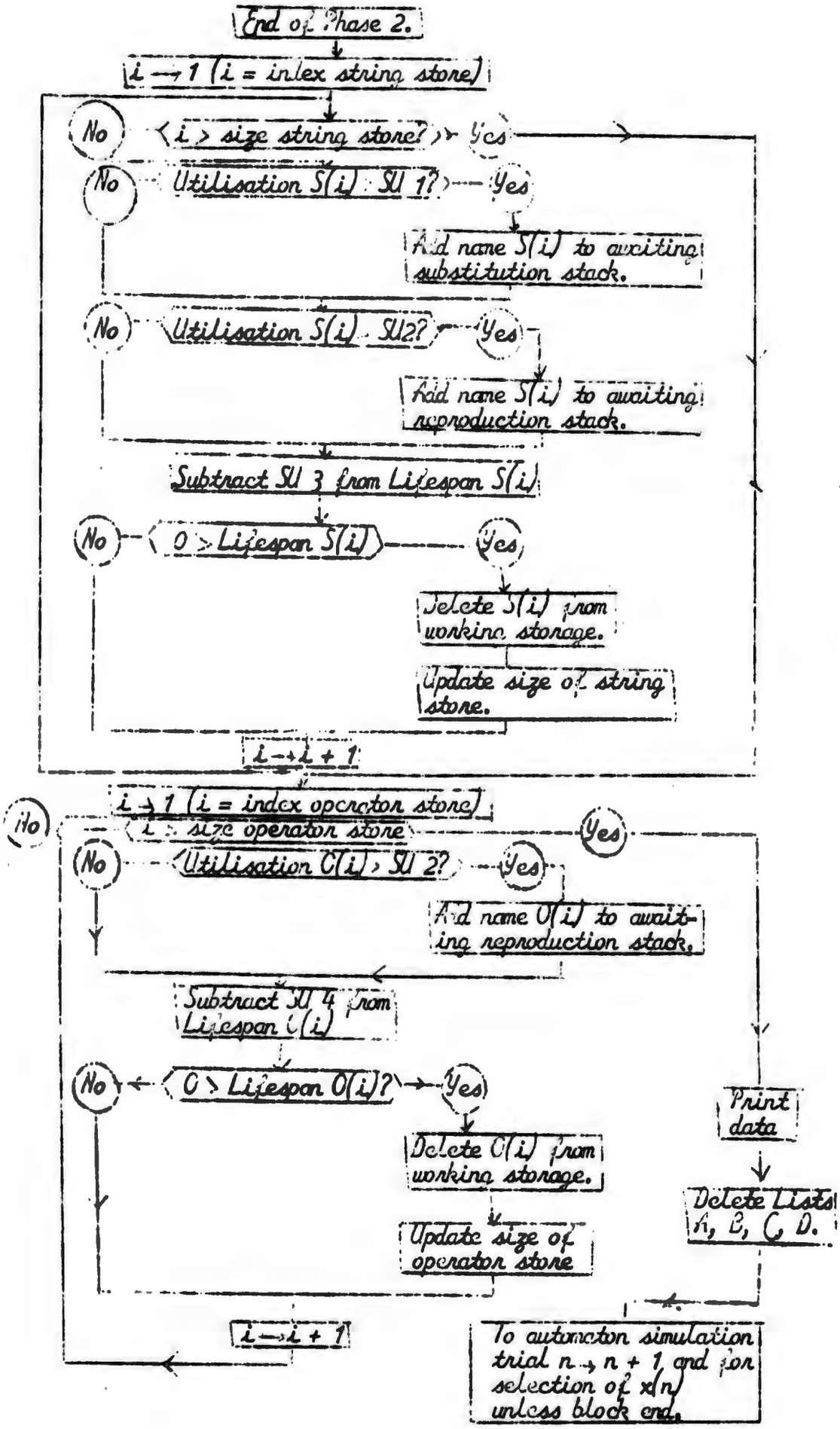
P.F.C. 9. Reproduction Procedure.

P.F.C. 10.



P.F.C. 10. Deletion Procedure.

P.F.C. 11.



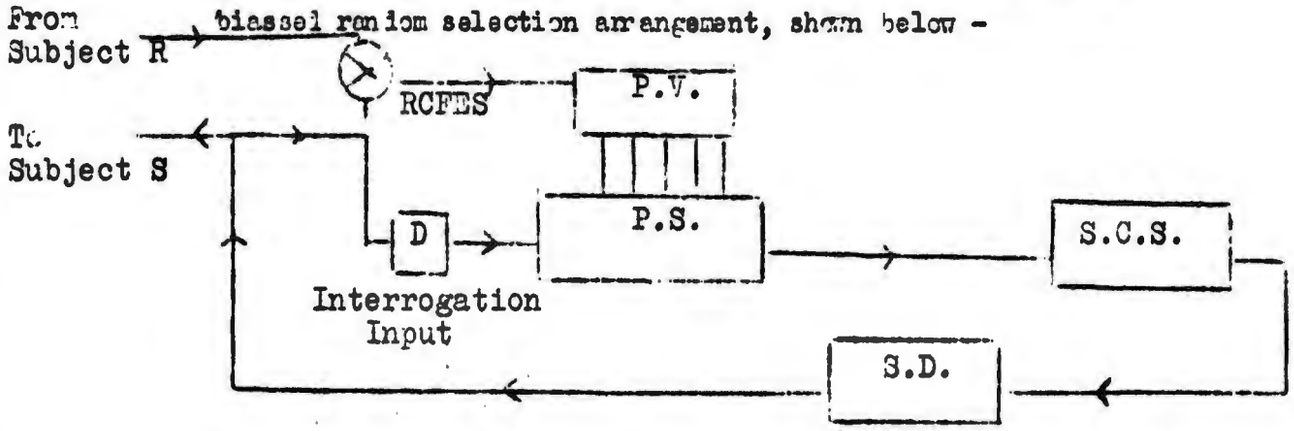
P.F.C. 11. Updating Procedure.

APPENDICES

Appendix.3.and Appendix.4.show representative data only.Complete records are available at this laboratory and these may be copied or inspected.

APPENDIX : 1

The stimulus generating procedure of Section 1.6.6. (II), in which the stimulus code tape is advanced contingent upon the occurrence of a completely correct response, is just one of several related procedures, all of which tend to homogenise the stimulus set (by rehearsing most often the most difficult problems) and for all of which it is possible to derive uncertainty measures. At the other extreme from the present procedure, there is the biassed random selection arrangement, shown below -



R = Response. S = Stimulus. R.C.F.E.S. = Response Comparator For Each Stimulus Separately. P.V. = Proficiency Vector. P.S. = Biassed Probabilistic Selector. S.C.S. = Stimulus Code Set. S.D. = Stimulus Derivation. D = Delay of Δt.

Using this arrangement, a block may be terminated either when each stimulus has received at least one correct response or when each stimulus has been presented on at least one occasion. Clearly, the former method is preferred for the present purpose.

In between these extremes, there are various procedures for forming a list of stimulus codes to be delivered in a subsequent block as a function of the subject's performance. Thus, if the response to stimulus x_0 is mistaken, then x_0 is repeated but <u>not</u> immediately afterwards. At least one stimulus is interpolated between x_0 and the repitition of x_0. The difficulty here is that further stimuli have to be added to the list towards the end of the block in order to provide something to interpolate.

APPENDIX:2

The Synthesis of Scoring Functions having the Shuford et al (Ref.6) Reproducing Property.

Consider a situation in which a student is asked to state the probabilities with which he estimates the items in a given set of possible answers are correct. For example, if there were three alternatives, he might state, $\frac{1}{3}$, $\frac{1}{3}$, $\frac{1}{3}$, if he had no idea which answer was correct, or he might state, 0, 1, 0, if he believed the second item was correct. Vectors in between would indicate varying degrees of uncertainty.

In a controlled teaching situation, the problem arises of awarding a score for the stated probability vector. This should be low for complete uncertainty and high for complete certainty and rectitude. Clearly, it is most useful if the student can be persuaded to output his actual belief vector in order to maximise his score. Therefore, assuming the student knows how the score is calculated, it is necessary to have a system which maximises his score only when his stated vector is the same as his actual belief vector.

Scoring systems or functions, which have this property, have been called Reproducing Scoring Systems (R.S.S.s) by Shuford et al (Ref.6).

This appendix outlines a method for synthesising R.S.S.s for the case of two choices. We have not been able to generalise the method to the case of n alternatives, but it has been possible to show that at least one of the functions synthesised by the 2-alternative method is generally applicable. This is the log scoring function (truncated for low arguments).

Definitions and Problem Statement

Let $\bar{P}$ be an n - dimensional vector with elements P_i for i = 12...n such that $\sum_{i=1}^{n} P_i = 1$.

Let $\bar{r}$ be another similar vector.

Thus, suppose $\bar{P}$ = Student's actual belief state.

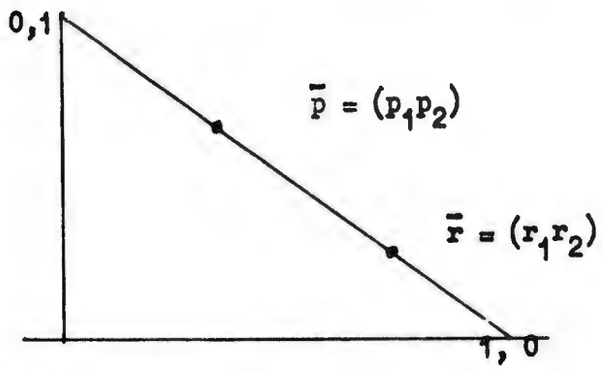
and $\bar{r}$ = Student's output belief state.

Then let F be a set of functions f_i defined so that $f_i(\bar{r})$ is the score the student receives if (a) the output $\bar{r}$ is his belief vector and (b) the i^{th} item on the list of n alternatives was the correct one.

Assume the subject knows how the f_i are computed. For any trial then, he can compute a Total Expected Score, TES, as follows:

$$TES = P_1 f_1(\bar{r}) + P_2 f_2(\bar{r}) + \ldots P_n f_n(\bar{r})$$

The problem can now be formulated as follows. How do we choose the f_i so that TES is maximised for any $\bar{P}$, if, and only if, $\bar{r} = \bar{p}$. Consider the case, when n = 2, illustrated below:



Then, $TES = P_1 f_1(\bar{r}) + P_2 f_2(\bar{r})$

At a maximum $\frac{\partial}{\partial t}$, TES = 0.

Whence $P_1 \frac{\partial}{\partial r_1} f_1(\bar{r}) + P_2 \frac{\partial}{\partial r_1} f_2(\bar{r}) = 0$

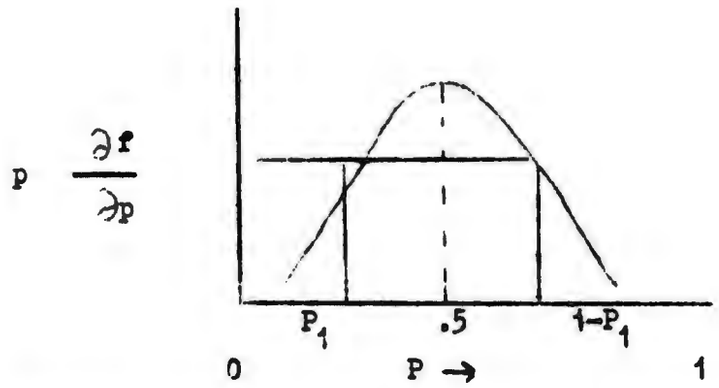
Or, $P_1 \frac{\partial}{\partial r_1} f_1(\bar{r}) = -P_2 \frac{\partial}{\partial r_1} f_2(\bar{r})$

Therefore, the requirement is that:

$$P_1 \frac{\partial}{\partial r_1} f_1(\bar{p}) = -P_2 \frac{\partial}{\partial r_2} f_2(\bar{p})$$

Now, in this case this corresponds to a simple symmetry condition (since $P_1 = 1 - P_2$) in the P $\frac{\partial f}{\partial p}$, P plane. It requires that the product of a value of P and the function derivative at that point be equal to minus the product of (1 - P) and the function derivative at <u>that</u> point, as can be seen from the diagram.

APPENDIX: 2



Theorem: Any function, $f(p)$, which is symmetrical about .5 in its P $\frac{\partial f}{\partial P}$, P plane is an R.S.S. for the two alternative case.

Proof: Let G (P $\frac{\partial f}{\partial P}$, P) be any symmetrical function as above.

Let P_1 be any value of x, so that $0 < p, < .5$ then

$$G\left(P \frac{\partial f}{\partial P}, P\right)_{p = p_1} = G\left(P \frac{\partial f}{\partial P}, P\right)_{p = 1 - p_1}$$

whence $$P \frac{\partial f}{\partial P}\Big|_{p = p_1} = -P \frac{\partial f}{\partial P}\Big|_{p = 1 - p_1}$$

Example 1

$$P \frac{\partial f}{\partial P} = \text{Const.}$$

Therefore, $f(p) = 1 + \log P \qquad .01 < P \leq 1$

$= 0 \qquad 0 \leq P \leq .01$

This truncated log function gives a score, f_n, varying between 0 and 1.

Example 2

$$P \frac{\partial f}{\partial P} = 2ap \qquad 0 \leq p < \tfrac{1}{2}$$

$$P \frac{\partial f}{\partial P} = 2a(1-p) \qquad \tfrac{1}{2} \leq x \leq 1$$

This gives a composite scoring function -

$f(P) = 2aP \qquad 0 \leq p < \frac{1}{2}$ (a-const)

and $f(P) = 2a(\log P - P) + 3.4a \qquad \frac{1}{2} \leq p \leq 1$

Both these examples yield R.S.S.s

The following analysis (after Brillouin) shows that the log function (example 1 above) is generally applicable. Consider the case when $p > .01$, then -

$$\text{TES} = P_1(1 + \log P_1) + P_2 (1 + \log P_2)$$
$$= \text{k} + \Sigma_{i=1}^{2} P_i \log P_i$$

Since it is a R.S.S. $\Sigma P_i \log P_i > \Sigma P_i \log r_i$ for $P_i \neq r_i$

We wish to show that this relation holds for $i > 2$

Let $r_i = p_i + q_i$

Then $\Sigma q_i = 0$

and $\Sigma p_i \log r_i = \Sigma_i p_i \log (p_i + q_i)$

$$= \Sigma_i p_i \log \left(p_i\left(1 + {}^{q_i}/_{p_i}\right)\right)$$

$$= \Sigma p_i \log p_i + \Sigma p_i \log \left(1 + {}^{q_i}/_{p_i}\right)$$

Now consider the functions ${}^{q}/_{p}$ and $\log (1 + {}^{q}/_{p})$.

These are equal if $q = 0$, otherwise ${}^{q}/_{p} > \log (1 + {}^{q}/_{p})$ wherever the log function is defined.

Then, ${}^{q}/_{p} \geq \log (1 + {}^{q}/_{p})$

Therefore, $\Sigma_i p_i \log p_i + \Sigma_i p_i \, {}^{q_i}/_{p_i} \geq \Sigma p_i \log p_i + \Sigma p_i \log (1 + {}^{q_i}/_{p_i})$

Therefore, $\Sigma_i p_i \log p_i \geq \Sigma_i p_i \log r_i$

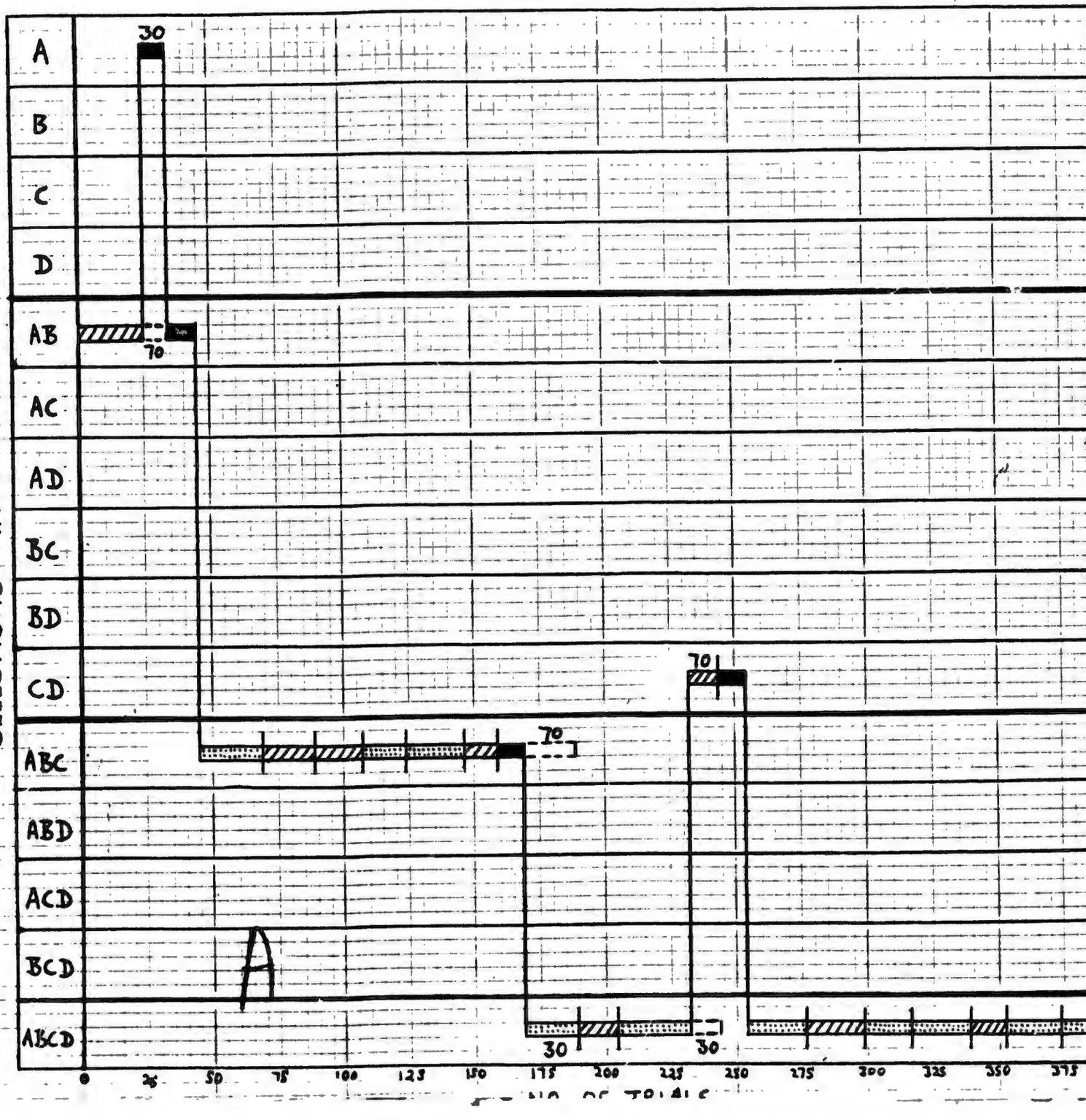
The equality holds when $p_i = r_i$

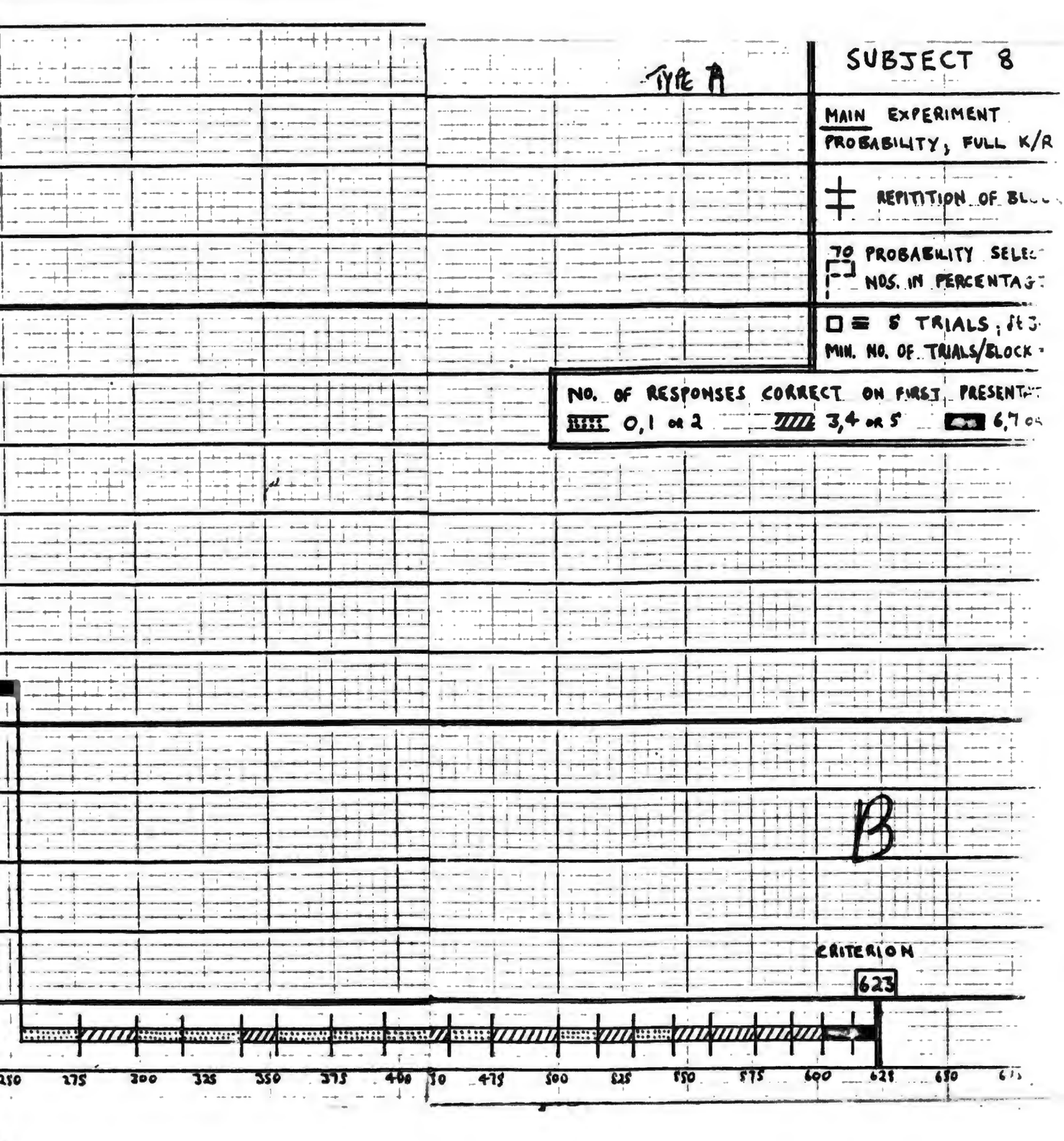
This analysis has shown that a log scoring function synthesised by the symmetry method for the two alternative case, can be applied to the general n alternative case.

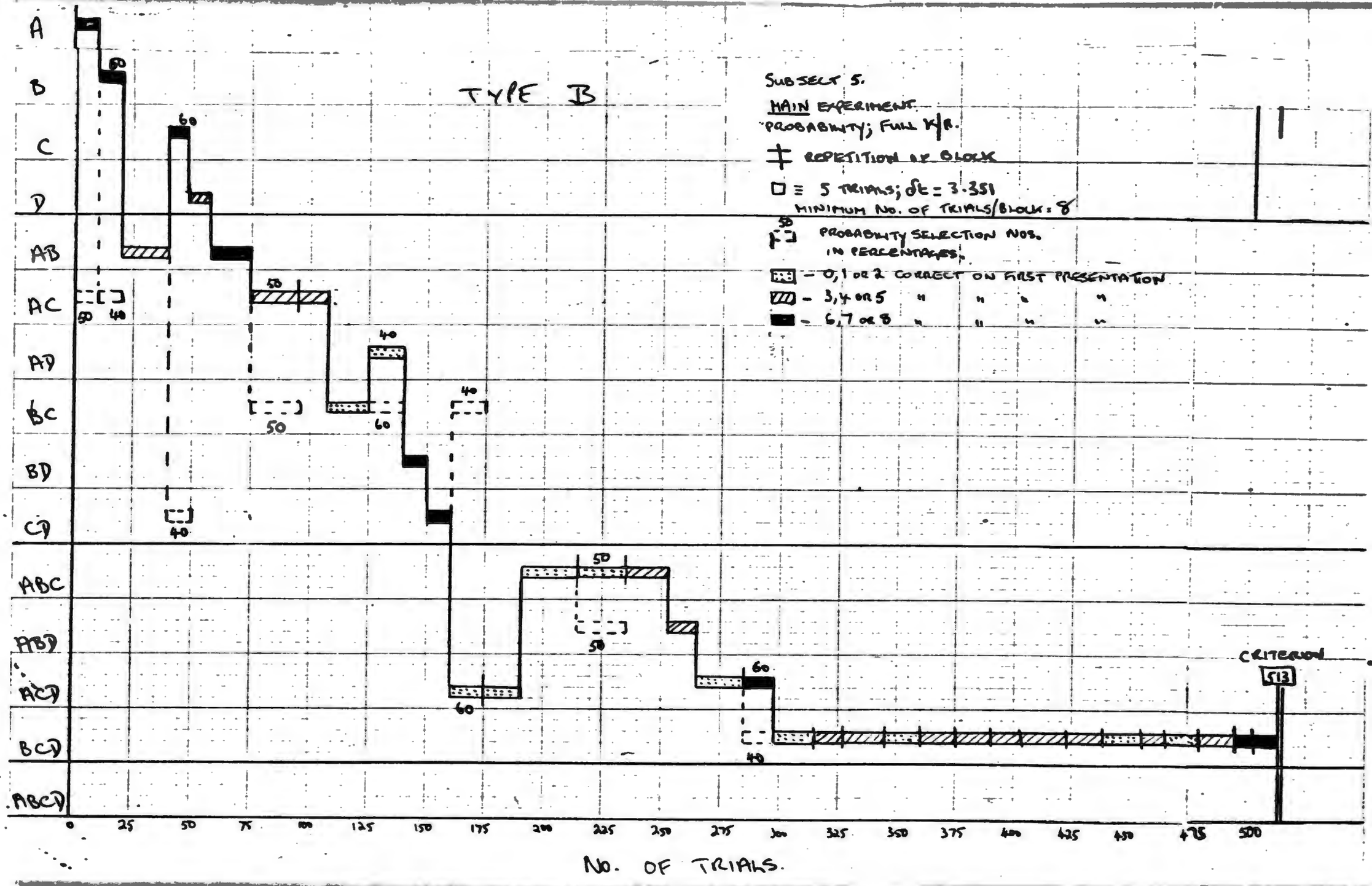
But it has not shown whether it is possible for all functions synthesised this way to be generalised.

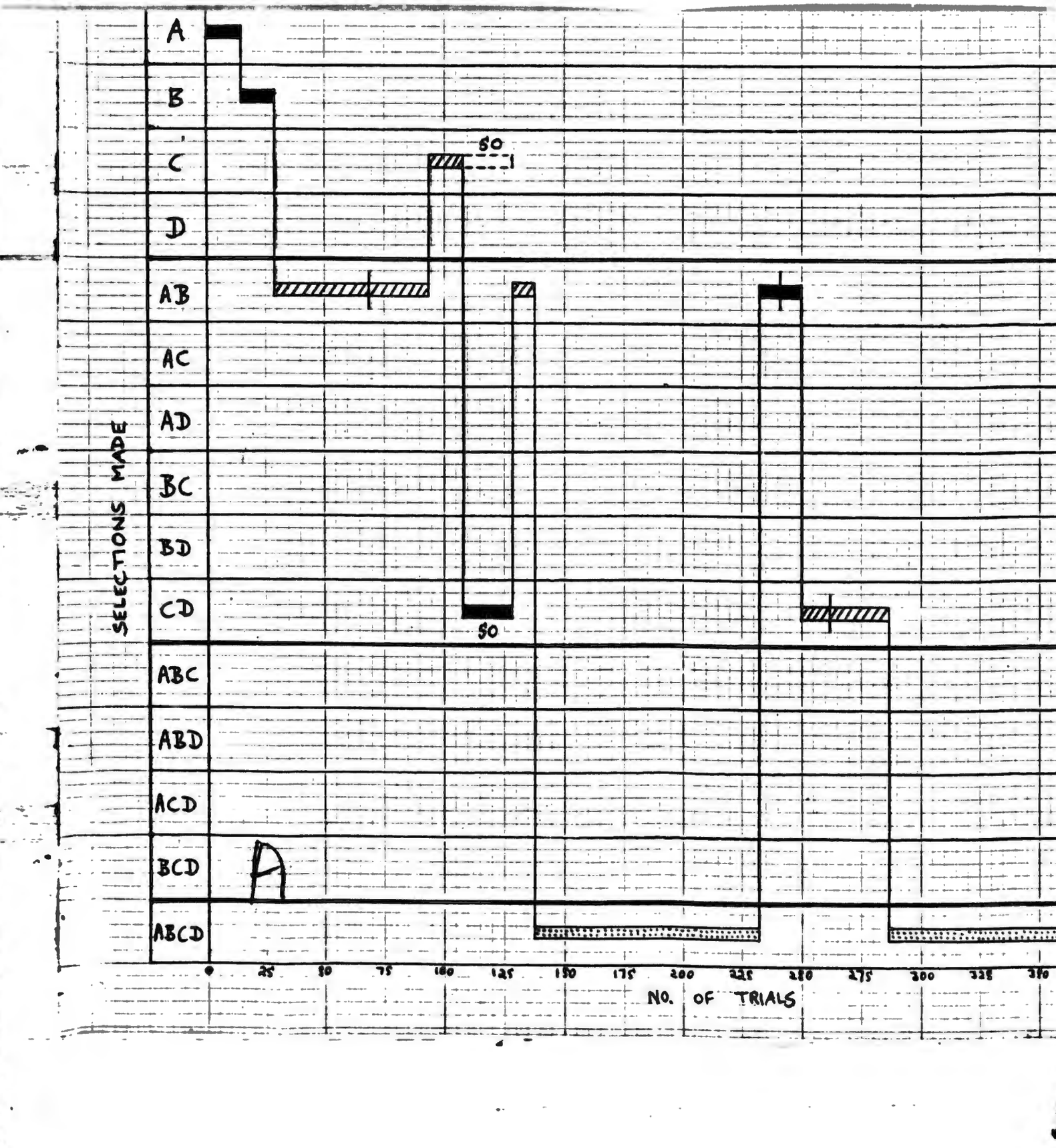
APPENDIX.3.

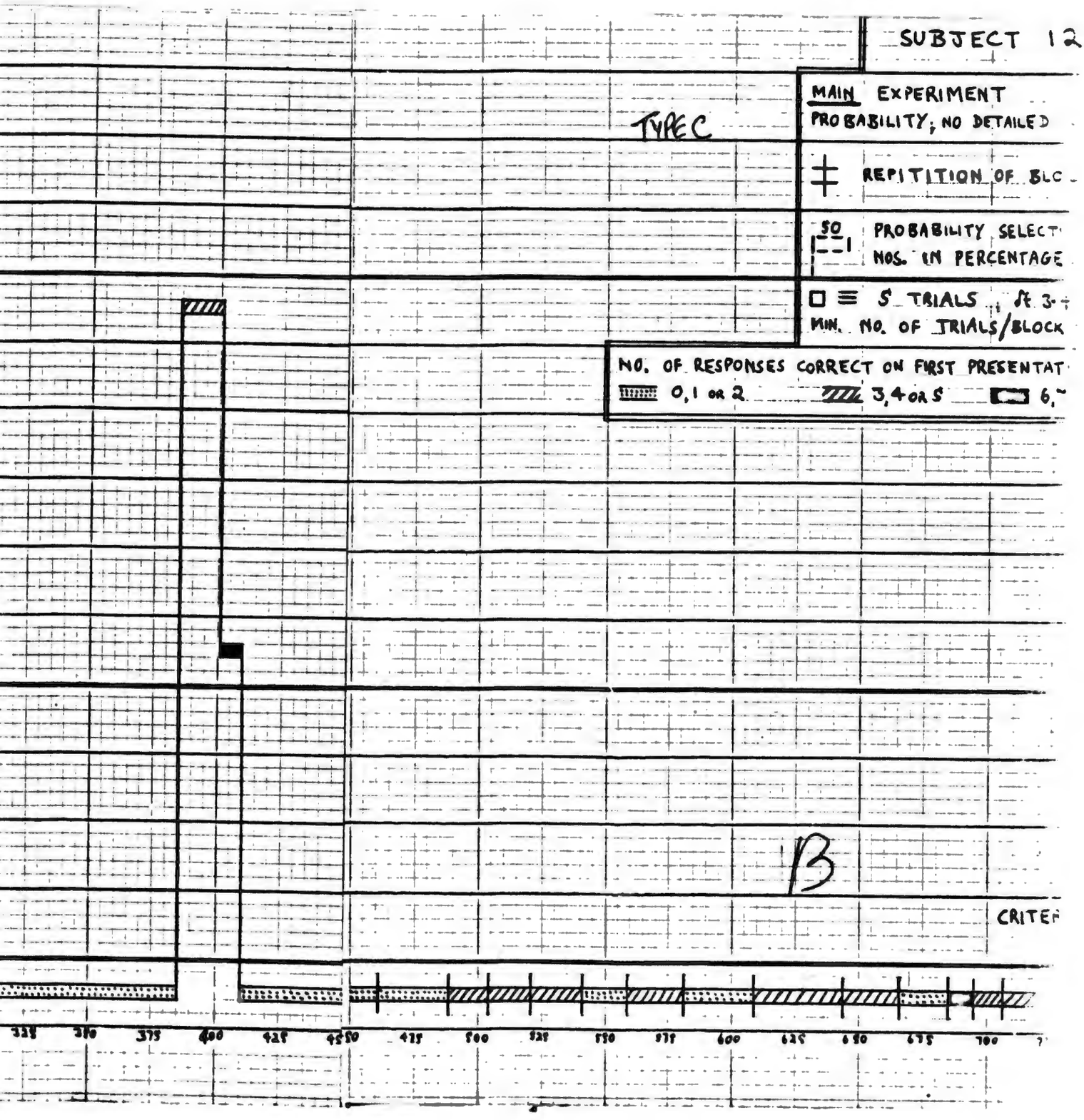
Typical data(learning curves)for subjects in the free learning condition.One example each of Type A Type B and Type C strategists.

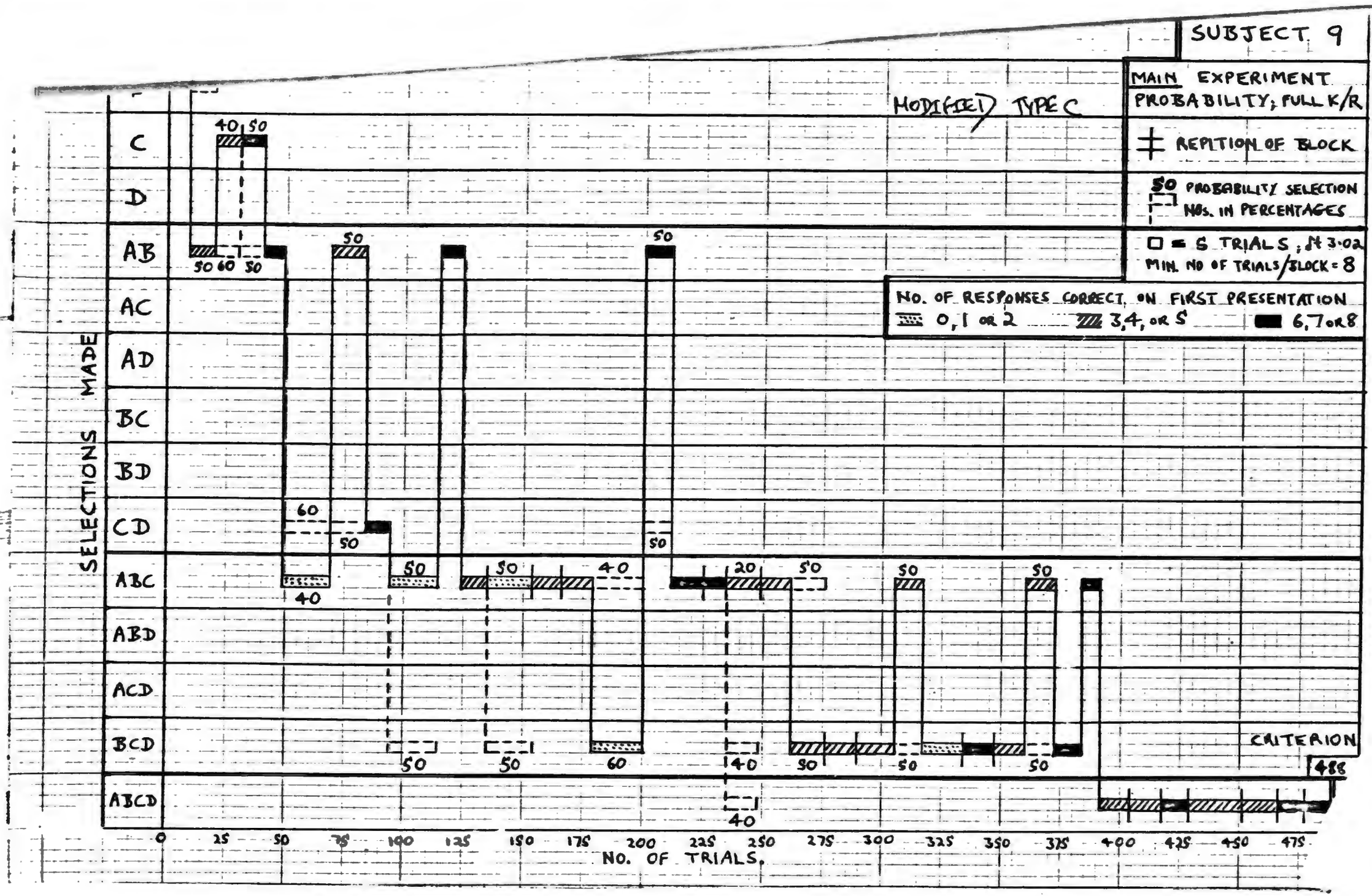

A
B
C
D
AB
AC
AD
BC
BD
CD
ABC
ABD
ACD
BCD
ABCD
30
70
70
70
30
30
A
SELECTIONS MADE
0
25
50
75
100
125
150
175
200
225
250
275
300
325
350
375


TYPE A
SUBJECT 8
MAIN EXPERIMENT
PROBABILITY; FULL K/R
REPITITION OF BL
70 PROBABILITY SELE
NOS. IN PERCENTAG
□ = 5 TRIALS;
MIN. NO. OF TRIALS/BLOCK
NO. OF RESPONSES CORRECT ON FIRST PRESENTA
0,1 or 2
3,4 or 5
6,7 o
B
CRITERION
623
250
275
300
325
350
375
400
450
475
500
525
550
575
600
625
650


TYPE B
SUBJECT 5.
MAIN EXPERIMENT.
PROBABILITY; FULL K/R.
‡ REPETITION OF BLOCK
□ ≡ 5 TRIALS; δt = 3·351
MINIMUM NO. OF TRIALS/BLOCK: 8
PROBABILITY SELECTION NOS. IN PERCENTAGES.
- 0, 1 OR 2 CORRECT ON FIRST PRESENTATION
- 3, 4 OR 5 " " " "
- 6, 7 OR 8 " " " "
A
B
C
D
AB
AC
AD
BC
BD
CD
ABC
ABD
ACD
BCD
ABCD
CRITERION
513
0 25 50 75 100 125 150 175 200 225 250 275 300 325 350 375 400 425 450 475 500
NO. OF TRIALS.

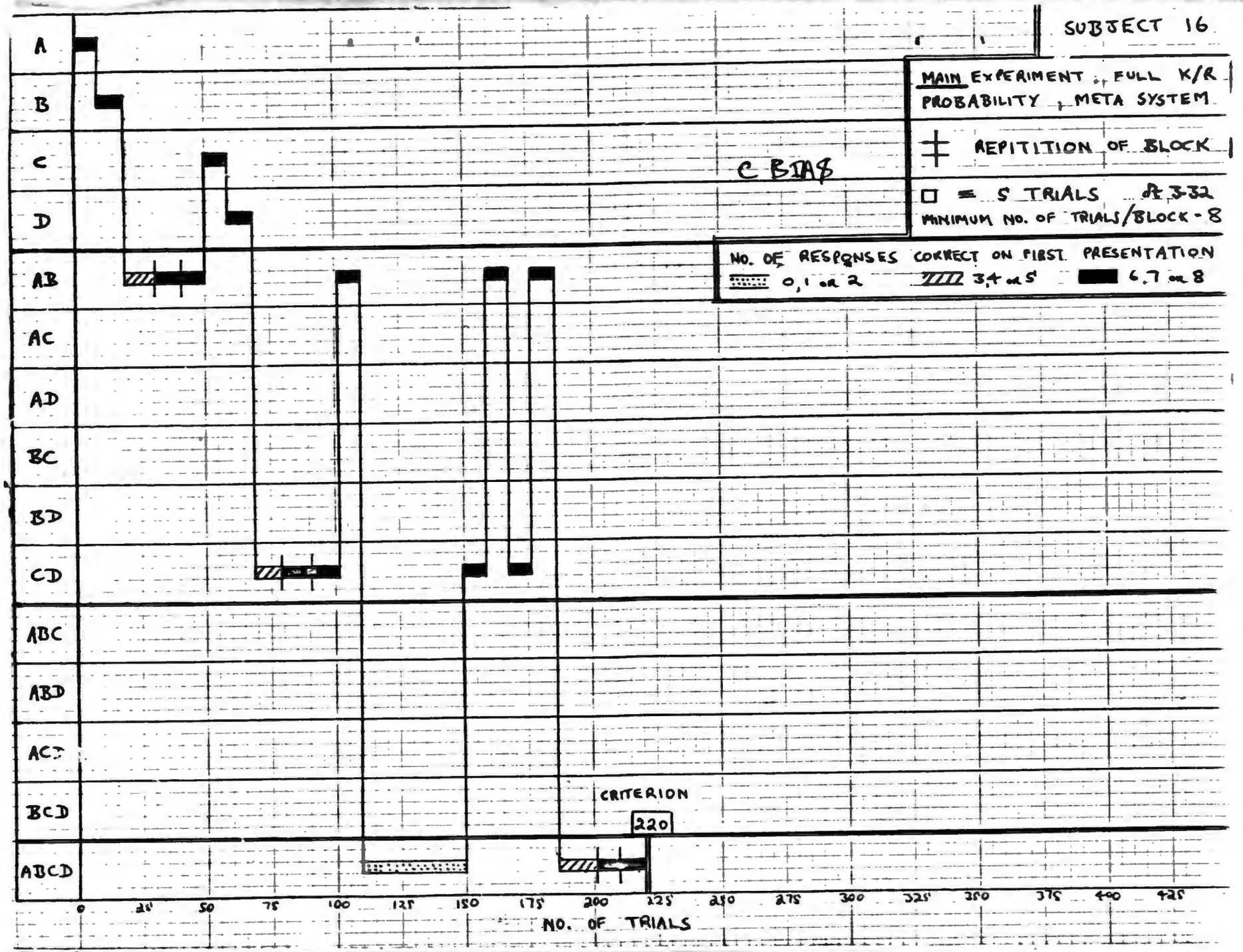

A
B
C
D
AB
AC
AD
BC
BD
CD
ABC
ABD
ACD
BCD
ABCD
SELECTIONS MADE
50
50
0
25
50
75
100
125
150
175
200
225
250
275
300
325
350
NO. OF TRIALS

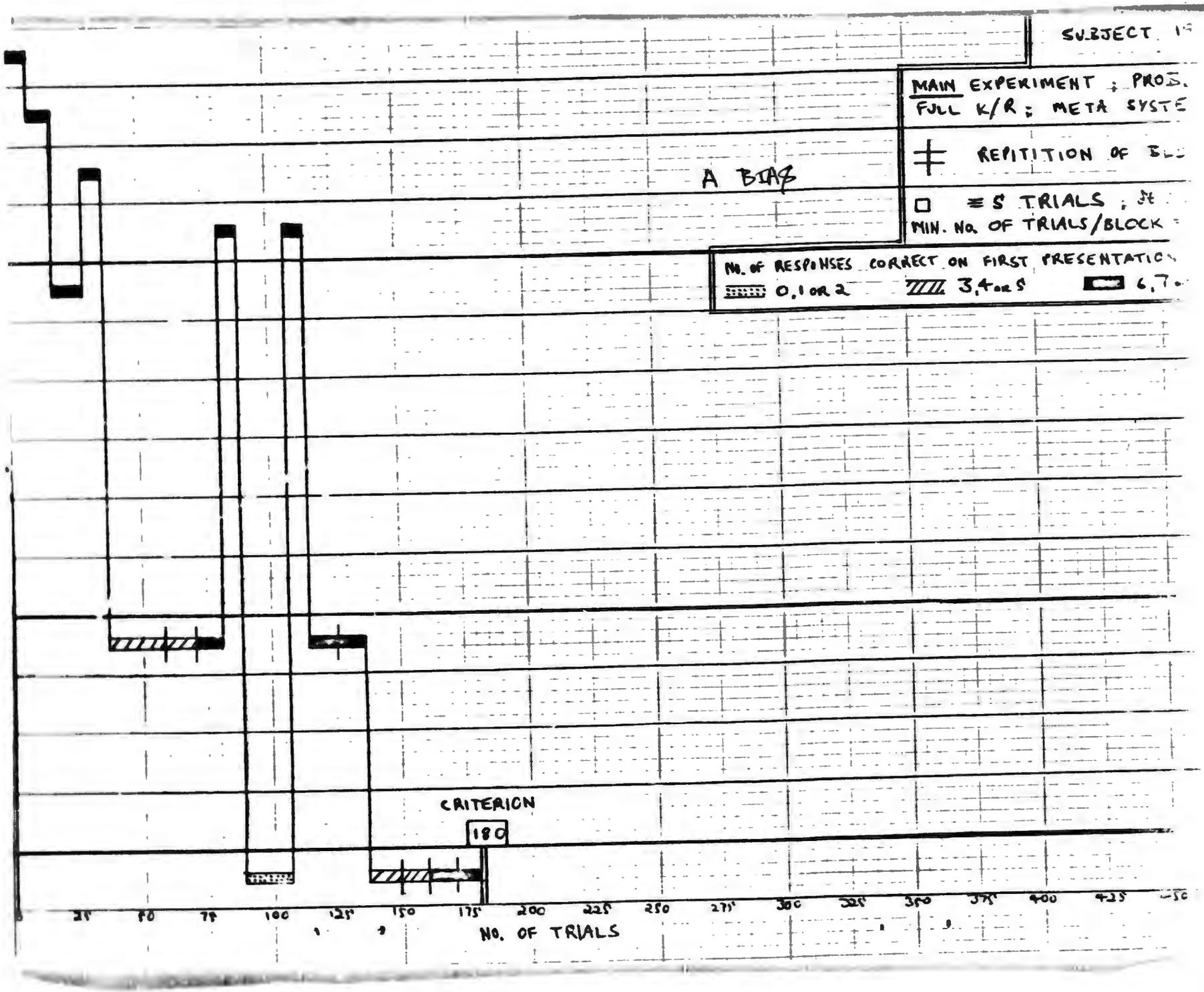

SUBJECT 12
MAIN EXPERIMENT
PROBABILITY; NO DETAILED
TYPE C
REPITITION OF BLC
50
PROBABILITY SELECT
NOS. IN PERCENTAGE
□ ≡ 5 TRIALS
MIN. NO. OF TRIALS/BLOCK
NO. OF RESPONSES CORRECT ON FIRST PRESENTAT
0, 1 or 2
3, 4 or 5
6,
B
CRITE
325
350
375
400
425
450
475
500
525
550
575
600
625
650
675
700


SUBJECT 9
MODIFIED TYPE C
MAIN EXPERIMENT
PROBABILITY; FULL K/R
= REPITION OF BLOCK
50 PROBABILITY SELECTION NOS. IN PERCENTAGES
□ = 5 TRIALS ; χ² 3.02
MIN. NO OF TRIALS/BLOCK = 8
NO. OF RESPONSES CORRECT, ON FIRST PRESENTATION
0, 1 OR 2
3, 4, OR 5
6, 7 OR 8
SELECTIONS MADE
C
D
AB
AC
AD
BC
BD
CD
ABC
ABD
ACD
BCD
ABCD
CRITERION
488
0 25 50 75 100 125 150 175 200 225 250 275 300 325 350 375 400 425 450 475
NO. OF TRIALS.

APPENDIX.4.

Typical data(learning curves)for subjects using the adaptive metasystem.One subject chooses to employ a Type C strategy at trial 50(subject 16). The other subject chooses to employ a Type A strategy at trial 35(subject 19).In common with all of the others run in the adaptive metasystem neither subject changed strategy or opted for exploration,though these possibilities are open.


SUBJECT 16
MAIN EXPERIMENT ; FULL K/R
PROBABILITY ; META SYSTEM
= REPITITION OF BLOCK
□ = 5 TRIALS
MINIMUM NO. OF TRIALS/BLOCK - 8
NO. OF RESPONSES CORRECT ON FIRST PRESENTATION
0, 1 or 2
3, 4 or 5
6, 7 or 8
C BIAS
A
B
C
D
AB
AC
AD
BC
BD
CD
ABC
ABD
ACD
BCD
ABCD
CRITERION
220
0
25
50
75
100
125
150
175
200
225
250
275
300
325
350
375
400
425
NO. OF TRIALS


SUBJECT
MAIN EXPERIMENT ; PRO
FULL K/R ; META SYST
≠ REPITITION OF
□ ≡ 5 TRIALS ; M
MIN. No. OF TRIALS/BLOCK
No. OF RESPONSES CORRECT ON FIRST PRESENTATIO
0, 1 OR 2
3, 4 OR 5
6, 7
A BIAS
CRITERION
18 O
25
50
75
100
125
150
175
200
225
250
275
300
325
350
375
400
425
No. OF TRIALS

Sequential Measure of Strategy Similarity

The following strategy sequences were used for comparisons of sequential similarity:

1. S's statement of a plan (selection sequence) made before the main experiment.
2. S's statement of a plan (selection sequence) made after the main experiment.
3. S's actual strategy (selection sequence) in pre-training.
4. S's actual strategy (selection sequence) in the main experiment.

Comparisons were made between 1 and 2, 1 and 4, 2 and 4 and 3 and 4.

For 1 and 2, S stated his overall strategy as the sequence of steps that would be generated, given the possibility of an error.

For 3 and 4, an equivalent "error-free" sequence was extracted from the actual sequence by these rules:

1. Omit any repetitions of steps.
2. Omit steps which are subpartitions of previous steps.

The similarity between two sequences (x and y) is calculated in the following way:

1. The steps of sequence x are numbered 1 to n from left to right, and are partitioned into subsequences of one, two, three and four attribute selections.
2. Sequence y is matched against sequence x as a series of subsequences, left to right. A step in y, which is identical to its equivalent in x, is assigned the same order number. A step in y, which is not identical to its equivalent in x, but is of the same subset (single, pair, triple), is assigned the order number of the step in x which is first in that subsequence. A step in y, which is not of the same subset of its equivalent in x, is given no score. The y scores are summed ($\angle y_x$).

3. A sum of scores, Σy^*_x, is obtained similarly, but sequence x is numbered 1 - n from right to left and matching is made from right to left.

4. The mean of Σy_x and Σy^*_x is divided by Σx and expressed as a percentage.

5. A similar % score is obtained by taking sequence y and matching sequence x against it to obtain Σx_y and Σx^*_y.

6. The measure of sequential similarity is then the mean of the two % scores obtained.

APPENDIX : 6

Can we regard the response latency patterns as events which signify the existence of stable particulate entities? By hypothesis, we can and some evidence for this point of view was provided by the integrity counts of 2.5.1. But what sort of stable entities do these events signify?? Well, by hypothesis, they signify the existence of problem solving programmes of various sorts that are brought from intermediate memory into immediate memory and used in solving problems (this point of view is elaborated in Section 5. For the moment, the crucial issue is that the stable entities reside in intermediate memory and are programmes).

Clearly, this hypothesis could be mistaken. For example, the subject might execute his problem solving programme, store the response names in immediate memory (perhaps in a rehearsal buffer) and output the responses with arbitrary latencies. One of our subjects tried to do this (with only a moderate degree of success) and has offered a very convincing picture of what he was up to (roughly, he adopted a special performance substrategy, which consisted in a transformation of the problem domain. As a result, he was able to execute the necessary response storage and to consciously control his latencies on about 25% of the trials). Incidentally, this subject is responsible for a very large proportion of the "scatter" in the otherwise immaculate "integrity counts" of 2.5.1. No other subjects were aware of such a substrategy, but this does not imply that they were innocent of using it.

However, it is easy to control against such a possibility. To do so, we need only randomise the value of δ t around the mean value assigned to it by the preliminary experiment and instruct the subject (1) It is best to produce the <u>entire</u> correct response (2) But it is next best to produce as many correct response <u>components</u> as you can in the randomised interval. These conditions should preclude the possibility of storing response component names in immediate memory and outputting the responses at leisure. Further, if the latency patterns retain their integrity under these circumstances, there is strong presumptive evidence that the stable entities are problem solving programmes.

Pilot experiments ,carried out along these lines,suggest that integrity is maintained and consequently that the grouping and stringing patterns are not merely artifacts due to a subject's whim.

If that is so,then we can proceed to examine the possible transitions or transformations of latency patterns and the influence exerted upon these transformations by stimuli and knowledge of results.Here,of course,stimuli and knowlidge of results appear in the role of contextual constraints.A stimulus such as $x_0 \in X_{ABCD}$ is"that which brings together certain existing programmes"(for example, programmes for responding y_1 to $x_1 \in X_{AB}$ and y_2 to $x_2 \in X_{CD}$). A stimulus is no longer "that which evokes a response".On the contrary,it sets the context for a response.

There is some evidence that the transformations are fairly regular.For example $y_1,y_2 \Rightarrow y_0$(a 4-tuple)if y_1 and y_2 are pairs.Further,some of the particulate entities are highly stable wheras others are not.In particular,groups are more stable(more"strongly bound")than strings which is illustrated by another commonplace sequence of transfor--mational events;

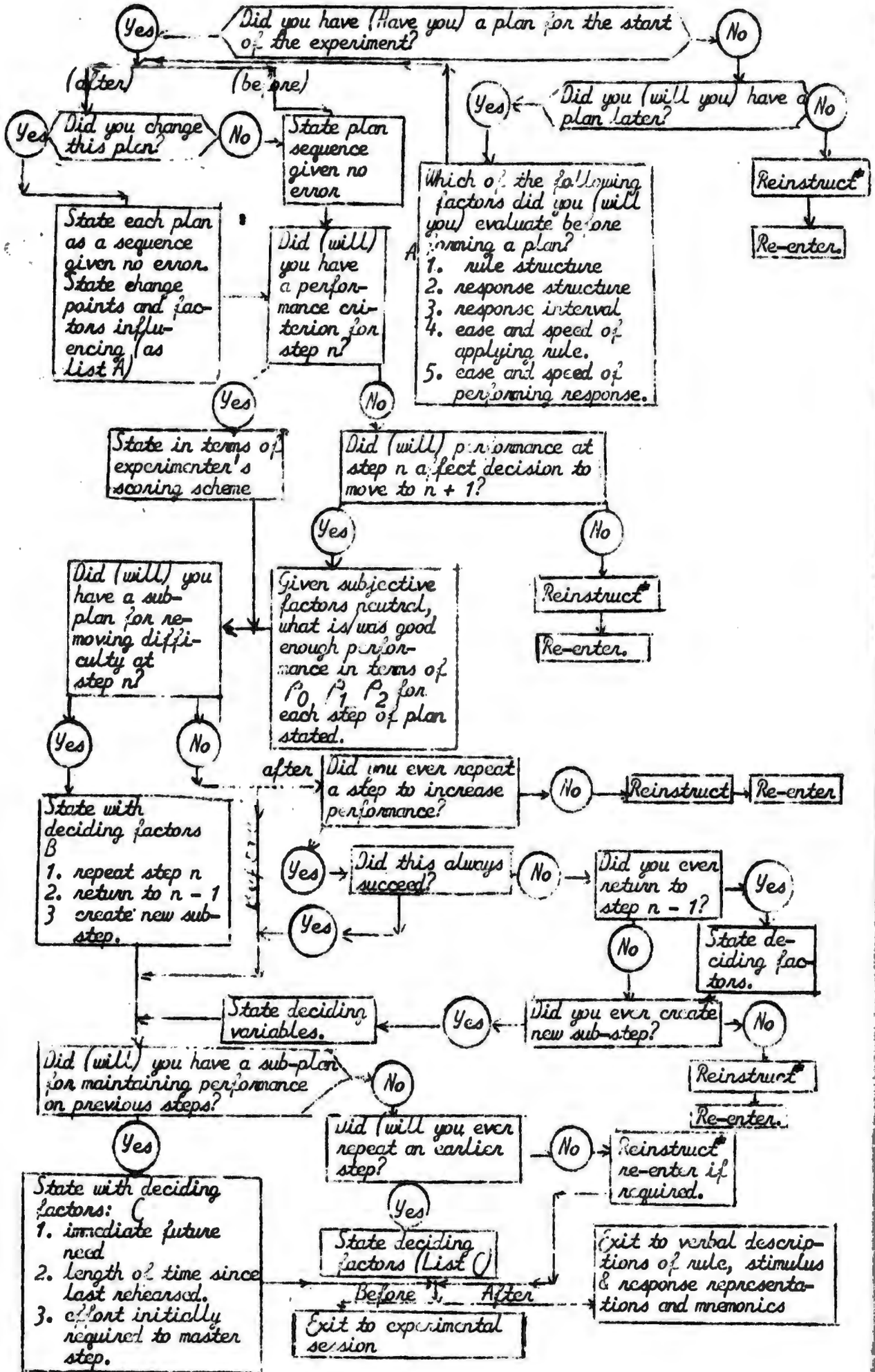
If $y_1 = y_a \frown y_b \frown y_c$ (a string) and if y_d is a single response component then,in the context of $x \in X_{ABCD}$

$$y_d,y_1 \Rightarrow y_d,y_a \frown y_c,y_b \Rightarrow y_b,y_b \frown y_d,y_a \Rightarrow y_a,y_b,y_c,y_d.$$

In all this,complete knowledge of results constitutes another contextual variable which modifies the form of the transformations and the stability of transient entities(such as the entity $y_d,y_a \frown y_c,y_b$).

Whether or not it is possible to work out a coherent "particle dynamics" is so far undetermined. But I believe it would be worth the effort of trying to do so and, perhaps, thereby, to achieve a "mental chemistry" at the level of goal directed problem solving programmes.

Appendix 7.



*'Reinstruct' represents rephrasing of questions & instructions where Subject has reached ambiguous or contradictory position

NOTE: 'List questions' were first asked as open ended queries. List presented where response ambiguous.

Measures of Performance and Uncertainty

Given a teaching situation in which block length continues until a specified number of correct responses is made, and in which a stimulus must be correctly responded to before another stimulus can be presented, a variety of performance measures might be used. Several are listed below -

Let N = No. of trials in a block.
Let S = No. of different stimuli presented during a block.
Let N_i = No. of presentations of ith stimulus i = 1 ... S.
Let C = No. of correct responses required to terminate block.
Let C_i = No. of correct responses to ith stimulus.
= No. of ith stimulus sequences.

Two different measures have been used, one giving an indication of the average doubt or uncertainty the subject has with regard to the stimuli in the block, and the other an indication of overall proficiency of performance in that block.

The first measure is defined as the average length of sequence of wrong responses. Since each sequence can be terminated only by a correct response, this measure, P_1, is given by -

$$P_1 = \frac{N - C}{C}$$

The second measure, P_2, is defined as the relative correct response frequency, given by -

$$P_2 = \frac{C}{N}$$

P_1 and P_2 are simply related. The mean length of a wrong response sequence, P_1, is the probability of a wrong response sequence, X, length of sequence.

Therefore, $P_1 = (1 - \frac{C}{N}) \frac{N}{C}$

$$= \frac{N}{C} (1 - P_2)$$

These measures have been used as reported in the main text. However, they are gross measures averaged over the block. More detailed measures, reflecting more accurately inter-stimulus performance, can be specified, as follows -

A measure like P_2 for proficiency for the i^{th} stimulus, is given by - $\frac{C_i}{N_i}$.

Therefore, the average over all stimuli is a proficiency measure -

$$P_3 = \frac{1}{S} \sum_{i=1}^{s} \frac{C_i}{N_i}$$

Note: $P_3 = P_2$ when $N_1 = N_2 \ldots\ldots N_S = \frac{N}{S}$

The $\frac{C_i}{N_i}$ are relative correct response frequencies and can, therefore, be combined in an entropy measure.

Using relative wrong response frequencies, an uncertainty measure, which for high values shows that all stimuli were equally difficult and for low values that some stimuli are more difficult than others, can be obtained.

Wrong response relative frequencies are -

$$\frac{N_i - C_i}{N_i}$$

Weighting these so that they sum to unity, we get the following weighted relative frequency -

$$\frac{\sum \frac{N_i - C_i}{N_i}}{\sum \left(\frac{\frac{N_i - C_i}{N_i}}{N_i} \right) N_i}$$

Therefore, the entropy of this weighted frequency distribution will be low when the $\frac{N_i - C_i}{N_i}$ are different, and high when they are similar. Calling this Differential Uncertainty, we get -

$$DU = \sum_{i=1}^{s} \frac{N_i - C_i}{KN_i} \log \frac{N_i - C_i}{KN_i} \quad \text{where } K = \Sigma \frac{N_i - C_i}{N_i}$$

This kind of measure can be interpreted as indicating the homogenity of block content.

1. Pask, G. and Lewis, B.N. "The Adaptively Controlled Instruction of a Transformation Skill". Programmed Learning, April 1967.

2. Pask, G. and Lewis, B.N. "The Use of a Null Point Method to Study the Acquisition of Simple and Complex Transformation Skills". British Journal of Mathematical and Statistical Psychology, Vol.21 Part 1, May 1968.

3. Pask, G. "Interaction between a Group of Subjects and an Adaptive Automaton to Produce a Self-Organising System for Decision Making" in Self-Organising Systems, edited by Yovitts, M.C., Jacobi, Y.T., and Goldstein, G.D. Spartan Books, 1961.

4. Baker, J.D. "The Uncertain Student and the Understanding Computer". NATO Symposium, Nice 1968.

5. Shuford, E.H. and Massengill, H.E. "Communication and Control in the Educational Process". AMRL-TR65-568. 1965.

6. Shuford, E.H., Massengill, H.E. and Albert, A. "Admissible Probability Measurement Procedures", Psychometrika, Vol.31. No.2., June, 1966.

7. Fiegenbaum, E.A. and Simon, H.A. "Elementary Perceiving and Memorising Machine", in Information Processing, edited by Popplewell, C.M., North Holland, 1962.

8. Atkinson, R.C. and Shriffin, R.M. "Human Memory, A Proposed System and its Control Processes". Technical Report, No. 710, Stanford University, March 1967, to be published in The Psychology of Learning and Motivation, Vol.II, edited by Spence, K.W. and Spence, J.T., Academic Press.

9. Pask, G. "Adaptive Teaching Systems" in Teaching Machines, edited by Austwick, K. Pergamon, 1966.

10. von Foerster, H. "Memory without Record" in Anatomy of Memory, edited by Kimble, D.P., Science and Behaviour books, 1965.

11. Mullen, C.L. and Pask, G. "The Method of Adaptively Controlled Psychological Learning Experiments" in Theory of Self Adaptive Control Systems, Plenum Press, 1966.

12. Mullen, C.L. "Recent Developments in the Theory and Practice of Adaptive Teaching Systems", in Aspects of Educational Technology, Methuen, 1968.

13. Kilmer, W.L., McCulloch, W.S., Blum, J., Craighill, E., Peterson, L. "On A Cybernetic Theory of the Reticular Formation" in Cybernetic Problems in Bionics, edited by Oestreicher, H.C. and Moore, D.H., Gordon and Breach, 1968.

14. Loefgren, L. "An Axiomatic Explanation of Complete Self Reproduction". Bulletin of Mathematical Biophysics. Vol.30, No.3, September, 1968.

15. Ashby, W.R. An Introduction to Cybernetics, Chapman and Hall, 1965.

16. Miller, G.N., Gallanter, E., Pribram, K.H., Plans and the Structure of Behaviour. Holt Dryden, 1960.

17. von Foerster, H. "What is Memory That it May Have Hindsight and Foresight as well", BCL Publication, 153, University of Illinois, 1968. To appear in Picture of the Brain Sciences, edited by Boroch, S.

18. Pask, G. "Man as a System that Needs to Learn", in Automaton Theory and Learning Systems, edited by Stewart, D.J., Academic Press, 1968.

Unclassified

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) System Research Ltd. 20 Hill Rise Richmond, Surrey, England	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

LEARNING STRATEGIES AND TEACHING STRATEGIES

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)
scientific: ; Final

5. AUTHOR(S) (First name, middle initial, last name)

Gordon Pask

6. REPORT DATE June 1969	7a. TOTAL NO. OF PAGES 180	7b. NO. OF REFS 18
8a. CONTRACT OR GRANT NO. F61052-67-C-0010 b. PROJECT NO. 9769-04 c. 6144501F d. 681304	9a. ORIGINATOR'S REPORT NUMBER(S) 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	

10. DISTRIBUTION STATEMENT

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Air Force Office of Scientific Research Directorate of Information Sciences Arlington, Va. 22209

13. ABSTRACT

Results are reported of human learning experiments and computer simulations. Experiments focused on (a) how a subject directs his attention and explores the problem environment created by the task to be learned, and (b) how the subject's learning process can be facilitated by conversational, or partially cooperative, teaching systems, in practice, CAI systems. In all experiments, the subject interacted with an automaton responsible for controlling the experimental environment. Under some conditions, the automaton was a procedural device and imposed behavioral constraints that represented rules of the task but permitted the subject freedom, particularly in his choice of a strategy. Under other conditions, the automaton was a simple adaptive teaching machine or a conversational machine. The simulation incorporated subprograms representing the subject and automaton respectively. The gross action of the metasystem is explained in terms of the theory of self-organizing systems. Different subjects exhibited different strategies for learning the same skill; the strategies may or may not be able to be fitted to aspects of mental organization of which the subject is often imperfectly aware. Results show that an adaptive metasystem is a better instructional instrument for the skill studied than either the attention-directing process in a free learning subject or a more restricted, single-strategy teaching system. The computer simulation of the learning process in the subject is a psychologically oriented artificial intelligence program, not a stochastic model. It is heterarchic in that its structure involves several separable but interacting hierarchic systems.

DD FORM 1 NOV 1473

Unclassified

Security Classification