THE UNIVERSITY OF MICHIGAN
COLLEGE OF LITERATURE, SCIENCE, AND THE ARTS
Computer and Communication Sciences Department

Technical Report

ADAPTIVE SEARCH USING SIMULATED EVOLUTION

Daniel Joseph Cavicchio, Jr.

with assistance from:

Department of Health, Education, and Welfare
National Institutes of Health
Grant No. GM-12236
Bethesda, Maryland

and

National Science Foundation
Grant No. GJ-519
Washington, D.C.

administered through:

OFFICE OF RESEARCH ADMINISTRATION ANN ARBOR

August 1970

ABSTRACT

ADAPTIVE SEARCH USING SIMULATED EVOLUTION

by
Daniel Joseph Cavicchio, Jr.

Chairman: John H. Holland

In this work we are concerned with studying the efficiency of adaptive systems. Adaptive systems can be interpreted within a formal framework so that adaptive plans can be viewed as search procedures for locating good devices in an extremely large set of devices.

Our major concern is the experimental development of some powerful and flexible adaptive plans. We proceed by first analyzing some previous work in adaptive systems within a formal framework. As a result we extract some of the common difficulties that these systems encounter. Of major importance are the improper use of feedback and inaccurate assumptions about the independence of components within a device.

Then we proceed to develop a set of adaptive plans, called reproductive plans, which overcome some of these difficulties. Reproductive plans operate by treating the search procedure as if it were an evolutionary process of finding the best organism for a certain environment. Devices are represented as chromosomes (strings). At each "generation" or time step a population of devices is tested

and each device is rewarded (duplicated) according to its performance. Then the duplicates are allowed to "mate" using a number of genetic-like operators to produce a modified population of devices. In this particular study devices are pattern recognition programs although they could be any set of (modifiable) procedures.

Much of our work is concerned with experimentally testing and improving the general reproductive plan to achieve fast and continuous adaptation. This is accomplished by testing the effects of and modifying (1) offspring selection schemes, (2) genetic operator probabilities and types, (3) the size of populations and chromosomes, and (4) other heuristic techniques to avoid false peaks and maintain effective adaptation. This work differs from similar work in the literature in that whole populations are evolved, many genetic operators are used, and efficiency and long term adaptation are stressed.

As a result we develop some very good reproductive plans which are superior to our original reproductive plans and which greatly outperform plans similar to those presented in the literature. We conclude that reproductive adaptive plans should prove to be valuable heuristic search procedures for extending much of the work done in artificial intelligence.

ACKNOWLEDGEMENTS

I wish to thank the members of my doctoral committee for their guidance in directing this research towards meaningful goals. My chairman, John Holland, was especially helpful in introducing me to adaptive systems, supplying up-to-date information on genetic mechanisms, and generally enrichening my graduate studies at The University of Michigan.

The members of The Logic of Computers Group provided a rich environment for the exchange of valuable ideas and interesting philosophies. Bernard Zeigler was especially helpful in the development of many of the formal aspects of this work. Ronald Brender, Daniel Frantz, and John Foy were consulted regularly for their computing knowledge. Mrs. Marilyn Abramson and Miss Janet McDougall gave willingly of their typing and drawing skills.

However, I must give special appreciation to people without which none of this work would have been possible. My parents sacrificed much to enable me to pursue my education to the Ph. D. level. I dedicate all of this effort to them. Finally I wish to thank my wife Janis for her constant encouragement.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF TABLES (Cont.)

LIST OF FIGURES

INDEX OF TERMS

[Including Page References]

Chapter 1 Introduction

The rapid expansion of the computer field in recent years has provoked an even more rapid expansion in man's desire to attack problems of great complexity. A decrease in computation time coupled with an increase in computer memory either immediately accessable in core or peripherally located, permits much of the computer's task to be performed in real time. As a result many previously tedious mathematical techniques, which are guaranteed analytically to provide solutions, can now be easily implemented. The fields of numerical analysis and mathematical statistics have reaped much of the benefit. However, a different class of techniques have also emerged which do not guarantee solutions, but which do offer increased information about the nature of certain problems and their approximate solutions. S. Lin has chosen to describe these *heuristic* techniques as practical, effective, probabilistic, flexible, and approximate as opposed to formal techniques which he considers theoretical, abstract, deterministic, rigid and yet precise [Lin, 1968].

Much of the work with heuristics falls in the domain of modelling or simulation since it extracts certain features of real systems in an attempt to predict future or unobserved aspects of these systems. There is, however, another branch which extracts aspects of real systems for purposes other than prediction. Although this latter work might result in a better understanding of the real system and a greater appreciation of how the extracted mechanisms serve the real system, these findings are not usually the principal concern. The

principal concern is with developing useful techniques which are applicable to a variety of problems. This is one branch of the field to which I would like to attach the ubiquitous terms "artificial intelligence".

As a few examples of these artificial intelligence systems we can cite Newell and Simon examining protocols of human problem solving behavior for their General Problem Solver [1963], Uhr and Vossler [1963] explaining their pattern recognition detectors in light of Hubel and Wiesel's experiments on the cat [1959], and Holland using techniques from evolution as a basis for an adaptive strategy [1969].

The artificial intelligence work that we will be concerned with in this study involves those systems which are adaptive or which respond to feedback information by changing their structure in one way or another in order to improve their performance. These systems are particularly well suited to the investigation of unsolved problems since they are generally flexible, exploring different variants of techniques and sometimes entirely different techniques in response to the feedback they receive. Furthermore, they are useful in situations where solutions to a number of problems are for the most part known, yet it is not always known exactly what problem is being confronted.

To facilitate the analysis of adaptive systems found in the literature, we should have a general theory of adaptive systems which could aid the process of isolating adaptive techniques. Given these techniques and some understanding of their usefulness, much of the yet unadaptive work in artificial intelligence could hopefully be extended.

The goal of this thesis is to investigate the analysis problem using a formal framework, develop a general adaptive plan which has a flexible structure, and demonstrate that this plan can actually achieve significant adaptation. In addition, we will demonstrate how this system could be applied to a variety of problems in artificial intelligence.

Upon examining the literature in adaptive systems one quickly encounters ambiguity. Most of the work fails to distinguish the adaptive mechanism from particular aspects of the task and the kind of feedback that the adaptive mechanisms use. As a result we find ourselves asking questions such as: What is actually adaptive? Where does the feedback come from? What is unknown about the system? What aspects of the system are particular to pattern recognition, problem solving, or game playing and what aspects are generally applicable to other problems? How can we compare one system to another? What is a useful criterion for evaluating different systems?

These questions do not have easy answers. However, we may start by looking at the various systems within a general framework. Chapter 2 will present such a framework and analyze some of the important literature in an attempt to extract general problems and techniques. Chapter 3 will then formulate an adaptive system which addresses itself to these problems. A set of plans called reproductive plans using techniques extracted from models of evolution and genetics will be described. In Chapter 4 we will investigate aspects of a suitable task with which to test the reproductive plans. First we will experiment with an

artificial task in order to get a feeling for some of the important aspects of reproducitve plans. Then we will proceed to apply our plan to a pattern recognition task which is more difficult and harder to analyze than the artificial task. In Chapter 5 we will apply some nonreproductive plans to the pattern recognition task in order to get an idea of what performance levels are reasonably obtainable using relatively unsophisticated plans. In Chapters 6 and 7 we will refine some of the basic operations of reproductive plans in order to make these plans more effective and flexible. Chapter 8 will show continuing improvement of the reproductive plan by the introduction of new techniques. Finally in Chapter 9 we will present some additional work being done with these schemes, suggest future research in this area, and summarize our work.

At each stage, a constant effort will be made to separate as much as possible those aspects which belong to the adaptive plan and those which are particular to the task under investigation. This attempt will begin by discussing reproductive plans in Chapter 3 before we have mentioned any particular task. In the end we will hopefully have convinced the reader of the need for a formal adaptive systems framework and of the value of reproductive plans in dealing with some of the problems in artificial intelligence.

Chapter 2 Analysis of Some Adaptive Systems

2.0 Adaptive Systems Terminology

In order to minimize ambiguity, we will adopt the following terminology. A *task* will refer to an attempt to solve a particular problem. The task is completed when the attempt is finished, whether or not the problem is solved. In artificial intelligence one often speaks, for example, of a game-playing task or a pattern recognition task. An *environment* may be described as a particular situation or set of circumstances in which a task is performed. Different opponents constitute different environments in a game-playing situation while various pattern categories induce a variety of environments in pattern recognition tasks. A *device* is a completely specified procedure (e.g., a computer program) capable of performing a given task in a given environment. A *representation of a device* is a well-defined method for coding a device. Given a code, one must be able to produce a unique device. An *adaptive plan* or *strategy* is a procedure which successively selects or generates devices, usually in accordance with certain performance measures. Finally, an *adaptive system* consists of adaptive plans, devices with a certain representation, environments and evaluation criteria. This will be more rigorously defined below.

2.1 An Adaptive Systems Framework

Let us examine briefly what aspects of adaptive systems are important enough to isolate in a formal framework. In general we should separately designate the actual task to be performed, the devices which are supposed to perform the task, and the adaptive plans which are supposed to choose the devices to be used. It should be noted that without

alternative devices we would need no adaptive plan and without alternative plans we would be unable to make comparisons.

The devices which are supposed to perform a given task can often be broken down into a portion which is common to all devices and a portion which varies from one device to another. This division is very important since some systems contain a learning routine which is distinct from adaptation. In many of these systems this learning algorithm is the same for all devices and is not changed by adaptation. Yet, in other systems it is the learning routine that *is* changed by adaptation. As one can surmise, confusion is only enhanced by using words like "learning" and "adapting", especially when they are interchanged in the literature. A formal framework should minimize this confusion by pin-pointing which portions of a device are modified by adaptation.

The source of feedback to the adaptive plan should also appear explicitly. Feedback provides various kinds of information about some unknown environment. One should know how much information is available to the adaptive plan and what possible range of environments this information is coming from. Actually the task and the task environment should be specified independently of the devices which confront the environment and the adaptive plans which choose among alternative devices.

Finally, one should have a well-defined criterion with which to rank different adaptive plans. This should not be confused with a similar measure which ranks the set of permissible devices. The criterion should facilitate the process of comparing different adaptive plans and even different adaptive systems once the environment, the set of devices, and the adaptive plans have been isolated.

Now let us quantify the discussion with a formal framework. An adaptive system is a quadruple of formal entities $< \tilde{\mathcal{A}}, \mathcal{E}, \mathcal{T}, \chi >$.

Definitions:

1) $\tilde{\mathcal{A}}$ is a triple $< \mathcal{A}, d, \mathcal{D} >$ which constitutes the representation of admissable devices.

$\mathcal{A}$ designates a possibly infinite set of well-defined codes for the devices belong to the set $\mathcal{D} \subseteq \mathcal{D}^*_T$.

The set $\mathcal{D}^*_T$ is the complete set of devices which are capable of performing a particular task, T. The function d associates a device with each code:

$$d: \mathcal{A} \rightarrow \mathcal{D}$$

Each code $A \in \mathcal{A}$ need only specify that portion of the device which may be changed through adaptation. We will often speak of device d(A) as belonging to the set $\mathcal{A}$ when no confusion will arise.

2) $\mathcal{E}$ designates a set of admissable (or possible) environments from which feedback will be obtained.

3) An adaptive plan $\tau \in \mathcal{T}$ is a quadruple $< \mathcal{M}, \mathcal{I}, m, \tilde{\tau} >$. $\mathcal{M}$ is the set of possible memory states for the plan τ and $\mathcal{I}$ is the set of possible feedback vectors. The function $\tilde{\tau}$ selects elements of $\mathcal{A}$ at each time step dependent upon previously selected elements, the current memory state, and the current input or feedback. To be more explicit, if:

A_t is the set of devices saved by the plan through time t, ($A_t \subseteq \mathscr{A}$)

M_t is the current memory storage of selected past information, ($M_t \in \mathscr{M}$)

I_t is the feedback vector for devices sampled at time t, ($I_t \in \mathscr{I}$) then:

$$\tilde{\tau}(A_t, M_t, I_t) = A_{t+1}$$

To complete the iteration, the function m updates the memory:

$$m(A_t, M_t, I_t,) = M_{t+1}$$

Actually the components of the vector I_t are the outputs of a number of functions acting upon A_t and E_t, the current environment. This will be considered in more detail below.

4) χ designates a criterion function which ranks elements of $\mathscr{T}$. In other words, χ specifies a particular measure of goodness with which one can compare elements of $\mathscr{T}$.

We will find the following function valuable in discussing the feedback vector and in examining various criteria:

Definition:

Let $\mathscr{R}$ designate the positive real numbers. Then μ will designate a ranking function with domain $\mathscr{E} \times \mathscr{A}$ and range $\mathscr{R}$.

$$\mu(E, A) = r \quad r \in \mathscr{R}$$

Given a particular $E \in \mathscr{E}$ we see that μ induces a ranking over the set $\mathscr{A}$. Since we will mainly be concerned with evaluating different elements of $\mathscr{A}$ in a particular environment $E \in \mathscr{E}$, we could refer to a particular function μ_E such that $\mu_E: \mathscr{A} \to \mathscr{R}$. However, the subscript E

will not appear when the environment considered is obvious or irrelevant. We will refer to $\mu(A)$, $A \in \mathcal{A}$ as the *utility* or *payoff* of A.

The function μ should appear explicitly or implicitly in all adaptive systems of interest since intuitively $\mu(A)$ represents how much each device $A \in \mathcal{A}$ is worth to the investigator. Different researchers might construct different μ's for a given set $\mathcal{A}$ in a given environment E, but for any particular μ it should hold that if $\mu(A_1) > \mu(A_2)$, $A_1, A_2 \in \mathcal{A}$, then one would always prefer A_1 over A_2 when concerned with the given task. The problem of defining a utility function which preserves this kind of preference relationship has been extensively considered by psychologists and economists.

Assuming that we have an adequate utility function, let us see how it could be used in setting up a criterion function χ. Basically χ expresses the worth of an element of $\mathcal{T}$ in the same manner that μ expresses the worth of an element of $\mathcal{A}$. The choice of χ will reflect the importance of certain factors as the set $\mathcal{A}$ is explored, insofar as χ will dictate which elements of $\mathcal{T}$ we should choose to use.

Let A_t^τ represent the set of devices selected by a plan $\tau \in \mathcal{T}$ at time t. Then an *accumulated utility criterion* would take the form:

$$\chi(\tau) = \sum_{t=1}^{T} \mu\ (A_t^\tau)$$

where T is some limit time. The function μ can be extended to subsets of A by taking the average or maximum over all elements.

A criterion of interest to artificial intelligence and search tasks is the *maximum utility criterion*:

$$\chi(\tau) = \underset{1 \leq t \leq f(\tau)}{\mathrm{MAX}} \left\{ \mu(A_t^\tau) \right\}$$

where $f(\tau)$ is some limit time at which the search terminates. Since we will assume that plans save a discrete number of devices at each time

step, $f(\tau)$ effectively puts a limit on the total number of samples permitted. In the simplest case $f(\tau)$ is constant for all τ. With such a criterion we see there is an opportunity cost involved in selecting elements of $\mathcal{A}$ insofar as only a specified number can be selected for consideration. Other f's might impose a limit depending on how well a plan seems to be doing.

The above criteria will prove useful in comparing plans which use essentially the same memory $\mathcal{M}$ and feedback information $\mathcal{I}$, differing only with respect to the transition functions $\tilde{\tau}$ and m. Plans which differ with respect to $\mathcal{M}$ and $\mathcal{I}$ can be properly compared only with a criterion that takes into account the differences in memory costs and feedback. Such criteria are hard to come by and the difficulties become even greater when one attempts to compare plans taken from different adaptive *systems*.

Now we should examine some aspects of the representation $\tilde{\mathcal{A}}$ and the effect of the representation on the plans $\mathcal{T}$. At first one might think it counterintuitive to have the function d map codes into devices. In the typical coding situation it is the object that is mapped into the code, with a suitable inverse function available to identify the object at a later time. In adaptive systems, however, the complete set of devices $\mathcal{D}_T^*$ is generally not well-defined or else very complicated to describe. The codes, whose image $\mathcal{D}$ under the function d is a subset of $\mathcal{D}_T^*$, are well-defined. Furthermore, d is not necessarily one-to-one. Thus we see that the representation $\tilde{\mathcal{A}}$ is an important aspect of adaptive systems since it restricts the range of devices that will be considered by the plans. This restriction is not necessarily bad; in fact, if the average utility of $\mathcal{D}$ is much greater than that of $\mathcal{D}_T^*$, the plan will probably benefit.

The composition of the set $\mathscr{A}$ is an even more important aspect of the representation than the $\mathscr{D}$ space that it induces. Since the set $\mathscr{D}$ is usually not available beforehand, the set $\mathscr{A}$ must be generated in a constructive manner. One possibility is to generate elements of $\mathscr{A}$ in a manner similar to the generation of well-formed formulae in logic. This would entail a primitive set of elements together with rules for generating new elements. Another method is to generate a co-ordinate system and define an element of $\mathscr{A}$ as a point in the resultant space. If the substitution instances along the co-ordinates are single real numbers ordered in the usual way (i.e., $\mathscr{A} \subseteq \mathscr{R}^n$), we shall call the space *numerical*. Otherwise the space is *nonnumerical*.

Let us look at some examples of the set $\mathscr{A}$. Suppose a device consists of a computer program with n subroutines. Each subroutine has an integer input from some finite range which specifies the number of times some iteration is performed. This input must be specified for all subroutines before the device is operative. Then the resulting numerical space $\mathscr{A}$ has n co-ordinates corresponding to the n subroutines. A code A is an n vector of integers.

Now suppose each subroutine were rewritten as many times as there were possible different integer inputs; these new subroutines would have different names but no input. Now a code would specify n symbolic names and the resultant $\mathscr{A}$ space is nonnumerical. In this case, however, the internal similarities between subroutines would be obscured. Certainly one would not choose this second representation for this task although the resultant set $\mathscr{D}$ is the same. Yet a program which can call one of a number of totally different subroutines at a particular point would possibly use the second representation.

The importance of a good representation becomes more evident when one considers the operation of a plan $\tau \in \mathcal{T}$. First of all, the memory $\mathcal{M}$ and input $\mathcal{I}$ are often dependent upon the particular representation. For example, a plan might receive input and save inferences about certain components of a device as represented in the code $A \in \mathcal{A}$. Inferences about a number and inferences about complete subroutines would probably take different forms. Secondly, the function $\tilde{\tau}$ will typically search a particular space by manipulating the coded descriptions constituting points. Gradient search methods serve as good examples.

To sum up, representation is important for two reasons: on the one hand, it may limit the maximum attainable utility by restricting the set of devices; in addition, it determines certain characteristics of the search space which will affect the performance of adaptive plans. Therefore, one must be careful not to blame the adaptive plan exclusively for poor performance that is a function of poor representation. Similarly, an unusual representation of devices could make a certain plan look very good, while over a range of other representations the plan or a suitable equivalent might only appear to be average.

Now let us return to the feedback vector $I \in \mathcal{I}$ and consider some of its elements. We will adopt the convention that the first element of I is always $\mu_E(A)$. If this element is not defined or used by a plan τ then we will designate the first element as ϕ, the empty element.

Definition: A plan $\tau \in \mathcal{T}$ is *first-order* if $I = [\mu_E(A)]$

Many times μ will have to be approximated. This is often the case when the task deals with only part of an environment each time it is

performed or when the device has a probabilistic element. In such cases, we will define $\mu'(A)$ to be the payoff for a particular running or trial of device A with a particular input from the environment. The actual $\mu(A)$ will be the average of the $\mu'(A)$ over a random sample of trials. Also, we will sometimes speak of μ-functions which are inversely related to the above described μ (e.g., error counting functions). This should cause no confusion since there still will exist a consistent ranking of the set $\mathscr{A}$.

Many researchers feel that their adaptive plans can extract and properly use more information than is given by the function μ. This extra information often reflects an evaluation of some component or substructure of a device $A \in \mathscr{A}$. Such components are usually co-ordinate substitution instances or primative elements which can be explicitly identified in the representation of a device A and ordered. Given an ordering of components $c_1,\ldots,c_i,\ldots,c_n$ we can define the set of functions,

$$u_{c_i} : \mathscr{A} \to \mathscr{R}, \quad i=1,\ldots,n$$

with the interpretation that u_{c_i} gives the worth of the ith component of a device. Each μ_{c_i} would occupy a position other than the first in the vector I.

Difficulties will often arise when a plan uses feedback from these μ_{c_i} functions. Let us suppose that the representation's code of a device consists of a vector of component substitution instances $\bar{c}_1,\ldots,\bar{c}_n$.

Definition: The function μ_{c_i} is *independent with respect to the representation* $c_1,\ldots,c_n$ *in an environment* E if there exists a function $\bar{\mu}_{c_i}$ such that for all $(\bar{c}_1,\ldots,\bar{c}_n)$

$$\mu_{c_i}(\bar{c}_1,\ldots,\bar{c}_n) = \bar{\mu}_{c_i}(\bar{c}_i)$$

when μ_{c_i} and $\bar{\mu}_{c_i}$ are evaluated in E.

Definition: A utility ranking function μ is *linear with respect to the representation* $c_1,\ldots,c_n$ *in an environment* E if there exists functions $\bar{\mu}_{c_1},\ldots,\bar{\mu}_{c_n}$ such that for all $(\bar{c}_1,\ldots,\bar{c}_n)$

$$\mu(\bar{c}_1,\ldots,\bar{c}_n) = \sum_{i=1}^{n} \bar{\mu}_{c_i}(\bar{c}_i)$$

The terms independent and linear will be used alone although there is always a representation and environment implied.

These concepts of independence and linearity will prove very useful in evaluating different adaptive systems. If a plan is to change a component of a device on the basis of a component utility function $\bar{\mu}_{c_i}$ then this function should be independent. An attempt to approximate a utility ranking function with a sum of independent component evaluation functions could fail for two reasons: the original utility function could be nonlinear due to interactions between components, or the component evaluation functions might not be the correct ones even if the original function is linear. Note that a linear ranking function implies the existence of independent component evaluation functions.

Now that we have some understanding for the utility function μ_E, we can more easily examine the set of environments $\mathscr{E}$. It is often difficult to extract particular environments for various adaptive systems. However, we can approach this problem by asking questions like, "What is it that changes the utility function, independent of a change in the researcher's desires?" or "What is unknown about the system?"

Now let us reword the questions asked in Chapter 1 in terms of our formal framework:

Original Question	Formal Interpretation
What is actually adaptive?	What is $\tilde{\mathcal{A}}$?
Where does the feedback come from? What is unknown about the system?	What are $\mathcal{I}$ and $\mathcal{E}$?
What aspects of the system are particular to pattern recognition, etc. and what aspects are generally applicable to other problems?	Is the representation $\tilde{\mathcal{A}}$ applicable to a variety of tasks? Are the plans $\mathcal{T}$ extremely dependent on $\tilde{\mathcal{A}}$?
How can we compare one system to another? What is a useful criterion for evaluating different systems?	What is a good χ? How can we adjust for different sets $\mathcal{M}$ and $\mathcal{I}$?

Before ending this section we should mention that there is one plan, the random search plan, which can be used by any adaptive system. The random search plan does not use any memory $\mathcal{M}$, feedback $\mathcal{I}$, or knowledge about the current devices A_t in selecting new devices. The device selection function $\tilde{\tau}$ merely selects a random subset of devices from some well-defined set $\mathcal{A}$. Therefore, it can be used regardless of the representation or feedback. A given plan may turn out to be only as good as, or even inferior to, the random search plan. The problem in this case might be that the information used by $\tilde{\tau}$ is irrelevant, $\tilde{\tau}$ is using information incorrectly, or $\tilde{\tau}$ is operating under severe restrictions (e.g., conducting only an extremely local search). Any of these situations may be the result of a poor representation. All adaptive systems should use the random search plan to establish, in effect, a 0-performance level in the ranking of adaptive plans. This would at least be a beginning in the

attempt to compare adaptive plans across different adaptive systems.

Now that we have presented a framework within which one can study adaptive systems, we will turn to some of the literature to see how well this framework aids the analysis of various works. Hopefully, it will help us pinpoint the difficulties that the systems under examination have encountered and provide some suggestions to overcome these difficulties. Also, we hope to be able to compare the plans of different systems as far as the representations will allow. Such comparisons might eventually enable one system to benefit from an adaptive plan that proved to be valuable to another system.

2.2 Important Literature in Adaptive Systems

In this section, we will briefly consider some of the research in the adaptive systems area. The general scheme of analysis will be as follows: first, the important aspects of the work will be qualitatively described including only as much detail as is needed to expose the significant features. In most cases this will result in a simplification of the work. Next, we will attempt to identify each of the elements $\mathcal{A}$, $\mathcal{E}$, $\mathcal{T}$, and χ. Finally, we will try to evaluate the work with respect to the representation chosen, possible alternative plans, and possible criterion.

The General Problems Solver (GPS) [Newell, 1963]

GPS operates in a system in which a task is specified as a goal. Elements of the system are well-defined objects which can be transformed into other objects by using various operators. In addition, there are means for detecting differences between objects and organizing information into subgoals. Three types of goals are possible: [Newell, 1963]

Transform object A into object B,

Reduce difference D between object A and object B

Apply operator Q to object A.

The objects and operators are considered to be given by the task whereas the differences are part of GPS. For example, if the task were symbolic logic, the objects would be well-formed logic formulae, the operators rules of inferences, and the differences expressions like "delete terms" or "change sign." GPS works towards a goal by generating subgoals which hopefully are easier and whose attainment will directly lead to the original goal.

Looked at from our formal framework one realizes that GPS as originally

presented is not an adaptive system but only an elaborate device. In line with the claim that the system is general, one might claim that at least it constitutues a set of devices; yet it is not completely obvious what devices, other than the one presented for symbolic logic, would look like.

Considering major tasks to be of the transform-objects types, the environment E for symbolic logic is the set of proofs for all theorems. A utility approximation function μ' could be defined as zero if the desired goal is not completed (possibly due to space and time limitations), or some positive number based on the difficulty of the goal and the effort required to complete the goal.

Newell, Shaw, and Simon realized the deficiencies of a nonadaptive system and presented a larger system to improve GPS [Newell, 1960]. One of the nice aspects of the larger system is that the adaptive plan is just the original GPS paradigm applied at a different level, i.e., with a different set of operators, objects and differences. Let us see how this works.

As mentioned above, a significant feature of the original GPS is the set of observable differences between objects. This is the portion that the authors chose to be adaptable. Therefore, an $A \varepsilon \mathscr{A}$ is a set of well-defined differences and the adaptive plan must find the "best" set. The plan is formulated as follows: let an element $A \varepsilon \mathscr{A}$ constitute an object for the upper level GPS. Operators at this level are functions mapping sets into sets. Examples are "delete a difference" or "modify a difference to give..."* Then there must be a set of upper level diffences on the set $\mathscr{A}$. The differences at this level identify certain good or bad aspects of each lower level difference set. The goals at

* The mechanism which generates and modifies differences is actually a very elaborate programming language which operates on lists and extracts "primitive discriminations".

the upper level are not as explicit as, "Transform object A into object A'" but rather take the form: "Find an object A' such that A' satisfies certain criteria (which A did not) and A' can be directly obtained from A with the available operators." Fortunately the "certain criteria" which really constitute the goal correspond closely to the aspects measured by the upper level difference set. Thus the upper level GPS works as follows: generate an initial set of differences which meet certain minimal criteria (e.g., consistency). Then directed by feedback from the upper level "differences", modify this set so as to improve it with respect to the ordered set of criteria.

In our formal framework the plan may be formulated as follows. While there is no significant memory $\mathcal{M}$, the feedback $\mathcal{I}$ is crucial. Consider a typical vector I_t:

$$I_t=[\{\phi,\mu'\},\ \mu^{a}_{c_i},\ldots,\ \mu^{a}_{c_n},\ldots,\ \mu^{j}_{c_n},\ \mu^{k},\ldots,\ \mu^{p}]^*$$

The pair $\{\phi,\mu'\}$ indicates the fact that sometimes the new set of differences were evaluated according to their performance on an elementary task while at other times they were evaluated only with respect to the "difference criteria". Most of the μ's output "1" or "0" to indicate that a certain criterion has or has not been satisfied. The function $\mu^{i}_{c_j}$ indicates the result of applying criterion i to difference c_j. An example of this type of criterion is, "Only one or a few operators should be relevant to each difference in the set." The criteria k etc., are applied to the whole set. For example, "The set of differences should be nearly orthogonal..." The order of the μ in the vector could indicate the importance of each criterion. A new set of differences is generated each time step by modifying the previously saved set. Thus the transition function $\tilde{\tau}$

* For convenience we will enter only the functions in the vector I_t with the interpretation that the elements of the vector are actually the outputs of these functions.

replaces a difference according to the criterion which must be satisfied. This transition function actually consists of the upper level GPS operators.

Although not explicitly mentioned it is fairly obvious how one might go about generating a whole set $\mathscr{T}$ of adaptive plans. Any change in the given criteria, the set of upper level differences or operators, or the method of generating new lower level differences would effectively change $\mathscr{I}$ and $\tilde{\tau}$. When this work was presented, the authors were still in the process of programming and testing the particular plan given, a task which is of considerable complexity. Therefore, the question of a criterion χ was not raised nor were alternative plans explored. However, one should have no trouble convincing himself that the plan given operates at a much better than chance level.

The representation of devices in $\mathscr{A}$, i.e., a GPS program, is probably responsible for much of its success. By explicitly isolating objects, operators, and differences at the lower level the device can recurse upon itself using the same operators and differences on subgoals. This representation also simplified the task of adaptation since the components (i.e., candidates for the adaptable portion) were explicitly defined. Another possible adaptable portion discussed by the authors is the table of connectives which specifies which operators are relevant to each difference. The same principles of representation were carried to the adaptive plan.

The adaptive plan does seem to have some limitations. For one thing, no where do the authors seem to suggest using a μ-function to get an accurate *overall* measure of each device. A μ which averages the μ' I have suggested above would be a necessary measure of a set of differences if only to check that each new set of differences does in fact perform better than the previous set on the given task. In a section on simulation

of adaptation, the authors suggest using simple problems to pinpoint deficiencies in the set of differences, but they do not mention an overall measure. Furthermore, it is not clear when and how many logic problems will be presented to each device. The main criticism is that without some directly task-oriented measure, performance will improve only with respect to the upper level criteria. The authors maintain that this results in directed, "intelligent" learning, yet there is no guarantee that these criteria will eventually lead to the best problem solving ability on the device level. Some of the criterion functions which examine each difference in turn are not independent with respect to the set of differences. Furthermore, the criteria do not measure independent aspects, so that by satisfying one criterion, a previously satisfied criterion may become unsatisfied. These interactions affect how efficiently the criteria direct adaptation.

Let us now summarize the elements of our framework:

$\mathcal{A}$ The set of all possible subsets of differences generated by a Difference Program Language

$E \varepsilon \mathcal{E}$ The set of theorems of logic

$\tau \varepsilon \mathcal{T}$ The method of selecting new subsets of differences as described above

μ Feedback designating the outcome of various criterion measures and sometimes an overall utility approximation

χ Not mentioned. The examples in the last section would be applicable, given alternative plans.

The Detector and Weight Problem

The next two works are grouped since the basic organization of each is similar; both are concerned with detector functions which are responsible for extracting information from the environment. With each detector is associated a weight which reflects the importance of that detector (e.g., a correlation co-efficient) or which relates that detector to a number of possible decisions (e.g., in pattern recognition). The adaptable portion of each device will consist of the set of detectors and/or weights. It is generally accepted that the detector problem (i.e., that of generating better sets of detectors) is more difficult than the weight problem, due essentially to the problems involved in generating and searching an adequate space of detectors.

Samuel's Checker Player [Samuel, 1963]

A device here corresponds to a checker playing program which can determine what move or decision is to be made at each stage of the game. A fixed set of detectors which measure aspects of board configurations such as piece advantage and center control are available. The output of these detectors is a positive or negative number which indicates the extent to which the player or opponent has the advantage with respect to each criteria. Each board configuration is evaluated using a linear polynomial. If $\{\theta_1,\ldots,\theta_k\}$ are the detectors and $\{a_1,\ldots,a_k\}$ are weights to indicate the importance of each detector, then $\sum_{i=1}^{k} a_i\theta_i$ constitutes the evaluation function.

The checker player operates as follows: at each move it looks ahead n steps in the game tree and evaluates all possible board configurations

at that point. It then uses a minimax basis to back up these values and select the best move. After the opponent has moved, it again looks ahead to see if the evaluation has changed as a result of the opponent's move or the additional board configurations evaluated with the look ahead. If the difference (called "delta", Δ) between the new evaluation and the old is sufficiently large, the weights are changed so as to make the two evaluations more consistent. Each weight is changed independently so as to slightly change the value of the entire polynomial.

Now we can use the formal framework. An element of $\mathcal{A}$ is a k-tuple of weights. $\mathcal{E}$ is the set of possible checker playing algorithms used by opponents. For analysis we shall assume that the opponent uses one fixed $E \in \mathcal{E}$ so that at each point his move is determined. The feedback vector has the following form:

$$I_t = [\mu', \mu_{c_1}, \ldots, \mu_{c_n}]$$

No μ' explicitly appears in Samuel's work, but the Δ factor plays a similar role. Since changes in the set of weights are possible at each time step (move), we see that no attempt is made to accurately assess μ over a variety of playing situations or games. It certainly would be too costly in terms of time and information loss to extend the time step to a whole game and just use win, lose or draw for payoff. However, it might help to extend the time step to a few moves and then make a change that is optimal with respect to all Δ observed. The μ_{c_i} conveys information about the sign of the ith term in the polynomial. This information determines how the ith weight will be changed. The memory $\mathcal{M}$ consists of a fixed number of previous board configurations and their evaluations with respect to each of the detectors. The function m replaces the oldest configuration with the current one. This memory is used in the look ahead process to determine Δ.

The particular $\tilde{\tau}$ used depends strongly on what conditions one required before the weights are changed, and exactly how they are changed. In this respect, Samuel experimented with different plans which depended upon how large a Δ had to be observed before a change was made. This was one way of side-stepping the utility approximation problem mentioned above; for example, a small Δ criterion would necessitate a change in weights just about every move resulting in erratic fluctuations in weight values. Other variations of the plan involved different types of change when Δ was negative and positive. This was done to avoid being fooled by bad play on the part of the opponent.

Since Samuel does in fact have a set of plans, we can ask what criterion is used to compare plans. One such criterion is stability. A plan which provides for a smooth, somewhat continuous change in weights is better than one that makes larger erratic changes. This criterion may be considered as a minimal one to insure some kind of convergence of the plan. However, eventually one is concerned with the best polynomial with respect to some payoff so the maximum utility criterion would be invoked:

$$\chi(\tau) = \underset{1 \leq t \leq f(\tau)}{\text{MAX}} \left\{ \mu(A_t^{\tau}) \right\}$$

The μ function might be the average of some inverse function of Δ and the function f might correspond to a time when the average value of observed Δ's no longer seems to be decreasing. This would mean that one could search for a better device as long as it seems that successive devices are in fact improving significantly. This latter criterion is used very often in artificial intelligence work.

Samuel's checker player is considered to be one of the best game

playing systems to date judging from its performance against professionals. Therefore, one might ask where the power of his system lies. For one thing, he has avoided the more difficult problem of finding good detectors. Yet, this did not greatly limit his program's performance since the detectors were formed after intensive consultation with expert checker players (not implying that this was an easy task). The linear polynomial certainly constitutes a significant part of the system. Its use did not result in very good play at the beginning nor end of the game, situations which call for specific identification of configurations rather than a general measure of various criteria. However, middle-game playing was much better.

Given the use of a linear polynomial, one can then ask, "How powerful is the adaptive plan which picks the 'best' weights?" or "How difficult is this task?" In my opinion, the plan does a good job of extracting and processing large amounts of information at each move of the game. On the other hand, a rather different plan which involves setting the weights so that the evaluation of various boards will be as consistent as possible with evaluations supplied by expert players proved also to converge on a rather good set of weights for the linear polynomial. Both of these plans illustrate interesting techniques for summarizing a large number of situations in a compact manner. However, their use is limited to truly linear situations.

There is one serious limitation in Samuel's evaluation technique. If the optimal use of all available detectors lies in some nonlinear combination, the linear polynomial scheme can only approximate this optimal performance. By his choice of $\mathscr{A}$ (i.e., all possible k-tuples of weights) and d (which generates the linear polynomial, the detectors, and the rest of the device) Samuel has restricted the set of devices $\mathscr{D}$ and possibly

the maximum performance level. Even if the best device did use a linear polynomial, there is no guarantee that Samuel's detector are the proper ones. In addition, Samuel's initial scheme for generating a new device (i.e., changing weights) implicitly involves an incorrect assumption of independence between detectors.

Samuel realized these problems and proposed an alternate method for using the detectors [Samuel, 1967]. Basically his signature table evaluation works as follows: subsets of detectors are grouped in the first level of tables, and records are kept as to which combinations of detectors values (i.e., outputs from the detectors) indicate a favorable board configuration as rated by experts. (See Figure 2.2.1) This method could potentially take into account all possible interactions between detectors since each set of detector values are considered independently. Therefore, a large output from a particular detector is not necessarily good or bad in all situations.

Processing information about all detector value combinations, even for a small set of detectors, would require too much memory and time; therefore, the value combinations of subsets are quantized before entering the next level of signature tables to be compared with outputs from other subsets. Therefore, the output of the first level tables, for example, could rank the boards on a nonlinear basis of the detectors involved as being very bad, bad, average, good, or very good. This process is continued at each level until a final evaluation is reached based upon some nonlinear interaction of all detector subsets.

Although the signature table organization is capable of extracting any interaction, difficulties still arise. First, the researcher must decide before hand upon the form of the table, i.e., which detectors to group, to what range he should quantize the values, how many levels, etc.

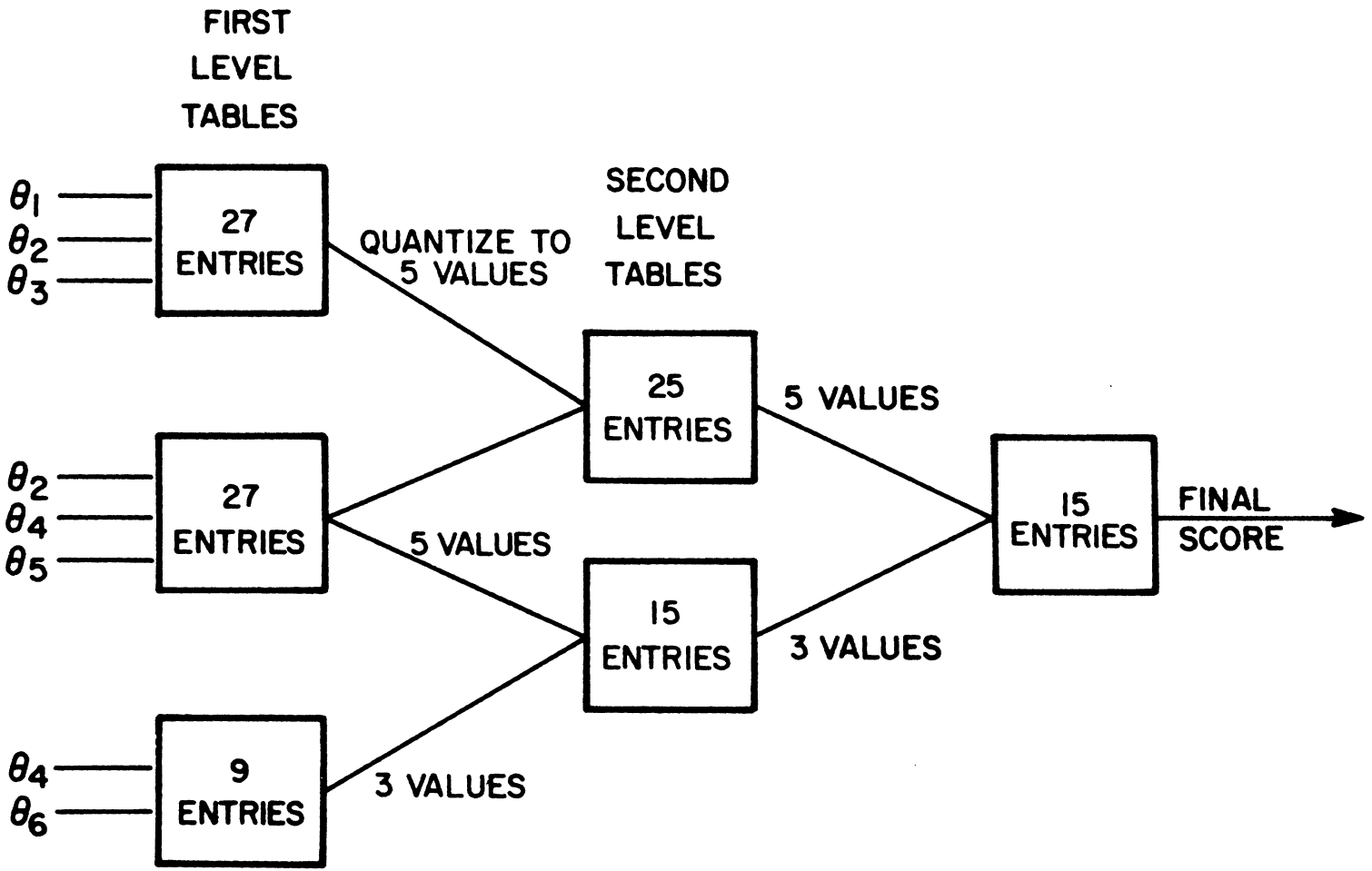


Figure 2.2.1 A 3-level signature table. Each θ can take three values, -1, 0, +1

Once these decisions are made, the power of the table is somewhat restricted. Referring to my example in Figure 2.2.1, a strong nonlinear interaction between θ_1 and θ_6 could not be extracted completely. Furthermore, each entry in each signature table is analogous to a weight in the linear polynomial scheme. Therefore, the signature table is not really operational until some information is gained about all combinations of detector *values* specified by the table. After investigation of many board configurations, some entries are still not different from their original dummy values. Therefore, they must be filled in by interpolation or some other linear approximation.

However, signature tables constitutes one of the first attempts to get around the problem of dependencies between components of a device. As such it gives us a refreshing look at an alternate representation. In addition, it did improve the checker player's performance. We mentioned the difficulties of the independence assumption with respect to GPS's differences and we will see them come up again.

The following summarizes the formal framework with respect to Samuel's systems:

LINEAR POLYNOMIAL

$\mathcal{A}$ The set of possible k-tuples of weights

$\mathcal{E}$ The set of possible playing methods of opponents

$\mathcal{T}$ The Δ-plans which modify the current set of weights or the "book learning" strategy which picks weights so that the polynomial reflects as well as possible the expert's evaluations observed up until that time

μ Basically Δ or the correlation of individual detectors with expert's decisions

χ Not mentioned explicitly. Stability and the maximum criterion are important. Also the ability to predict expert's decisions.

SIGNATURE TABLES

$\mathscr{A}$ The set of all possible sets of signature table entries

$\mathscr{E}$ The set of possible playing methods of opponents

τ The process of modifying entries similar to the "book learning" process mentioned above

μ Expert's evaluations of board configurations

χ The ability to predict expert's decisions.

It should be noted that the size of $\mathscr{A}$ in the signature table system makes adaptation slower and necessitates the use of interpolation and other approximation schemes.

Much of the power of Samuel's system lies in his detectors. Unfortunate these were hand tailored for the task, a process which at present is still an art. Some systems have attacked the detector problem as evidenced in our next example. However, in this example the task is somewhat artificial and the problem of *generating* detectors nonexistent.

Klopf's Pattern Recognition [Klopf, 1965]

A. W. Klopf developed a pattern recognition system which nicely fits into our formal framework.* In addition its structure closely resembles Samuel's linear polynomial.

The task is to uniquely identify each of 2^n patterns where a pattern is specified by n binary inputs to the device. In this study, the main

* Klopf's system is called evolutionary, but to avoid confusion with later discussion it will not be analyzed with the language the author uses since the evolutionary concepts are minimal.

concern is not with finding a good representation for pattern recognition, but with finding the best device given a particular representation. In other words, Klopf is interested in studying various adaptive plans.

A device consists of a set of m randomly generated functions, whose input is the n binary numbers and whose output is a single number. A new function can be generated by specifying 2^n new random numbers to make up the range of the new function. The output of each function is multiplied by a weight and the resultant values summed (See Figure 2.2.2). One can see that different input vectors will generally result in different values of the weighted sum, Z. Certain values of Z are arbitrarily chosen to represent each of the 2^n patterns. Therefore, the weights must be adjusted so as to correctly identify as many patterns as possible, i.e., the output Z for each pattern input should come as close as possible to the randomly chosen Z value for each pattern.

Unlike Samuel, Klopf did not include the weight adjustment problem as part of his adaptation. Instead, he used an available routine which always produced the optimal set of weights for this task. The existence of such a routine seems to imply that the weight setting problem is generally not too difficult. Klopf's adaptation involved finding a good set of random functions. The basic adaptive plan is as follows: generate an initial set of random functions. Evaluate each function using prescribed μ_{c_i} evaluation functions. Replace a certain number of the worst functions with randomly chosen new functions. Repeat the procedure until performance as measured by a utility function ceases to improve significantly. The μ used in the criterion χ was an error counting function, where the error is the difference between the observed Z and the correct Z. Klopf used the maximum utility criterion, stopping adaptation when the

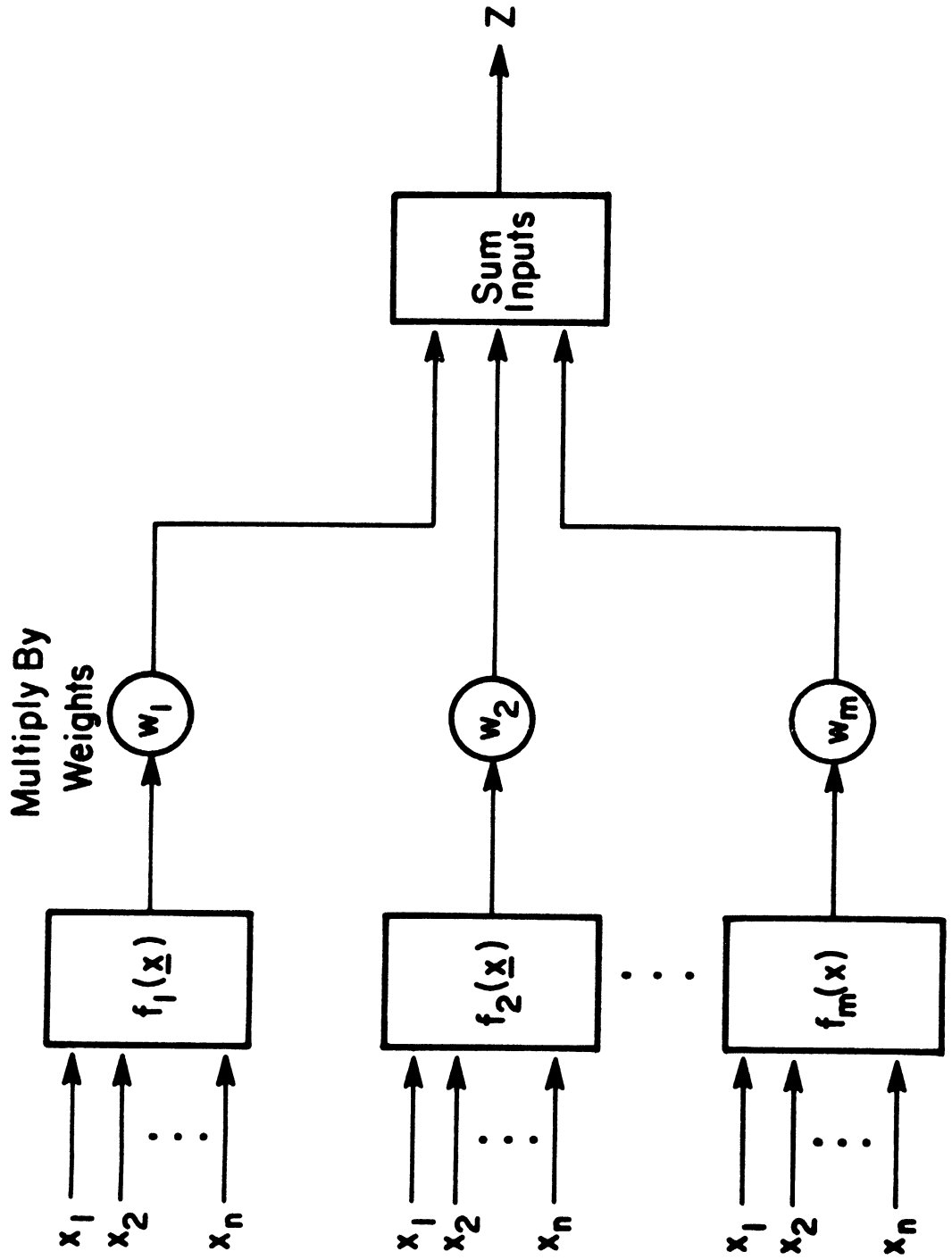


Figure 2.2.2 Klopf's pattern recognition system

error ceased decreasing. He varied his plans by varying the μ_{c_i} evaluator functions and the schedule which specified how many functions were to be replaced at each step. His results included a specification of the best μ_{c_i} and best replacement schedule along with statistical estimates to show that his best plan performed much better than random search.

This work is representative of much of the work done in adaptive pattern recognition and adaptive systems. Although there is a perfectly good overall utility function available, Klopf does not use it. Instead he uses component evaluation functions which are not independent. Yet, he uses these functions as if they were independent. For example, one type of μ_{c_i} function measured correlations between the random function outputs and the final output Z, while another attached utility to the size of the weight associated with each random function. In addition, Klopf does not check to make sure that the inferior functions are not replaced by even worse functions. Such a check is especially necessary later in adaptation when gains are hard to come by and should be preserved.

The independence assumption is not generally valid. However, a scheme very similar to Klopf's is used by Uhr and Vossler in their adaptive pattern recognition program [1963]. They try to simultaneously modify weights and detectors, yet they evaluate each detector as if it alone were responsible for the recognition task. Although a plan based on an independence assumption might sometimes be a good approximation to the best plan, it seems that some other method should be available to search the set $\mathscr{A}$ in a more sophisticated manner, especially when independence assumptions break down.

To summarize Klopf's system we have:

$\mathcal{A}$ All possible sets of n random functions

$\mathcal{E}$ All possible name (number) assignments associated with 2^n "patterns"

$\mathcal{T}$ The strategies for modifying the existing function set. These plans differ with respect to the evaluation techniques used and the number of functions replaced (alternatively the size of the step taken in the search space). There is no memory $\mathcal{M}$ involved.

μ The results of certain detector evaluation criteria

χ The maximum criterion.

Fogel *et al.'s* Evolution [Fogel, 1966]

Fogel, Owens and Walsh wrote a book whose main purpose was to study a particular adaptive system. Their claim was that the system would have general applicability rather than be tailored to a particular task. However, the authors just about defeated their own goals by a poor choice for the representation of their devices and a limited view of the environment.

Their devices consisted of finite state machines. A finite state machine deals with a finite input and output alphabet. Given an initial state and a sequence of input symbols, the machine will move through a sequence of internal states and output a symbol dependent upon each state transition it makes. Figure 2.2.3 is an example of a three state machine with 0,1 as input symbols and α,β output symbols associated with each transition. The set of all finite state machines is obviously infinite.

A particular environment confronting the machines is some relation defined on an infinite sequence of input symbols. A simple example is the repetition of some subsequence. The general task is to find a device

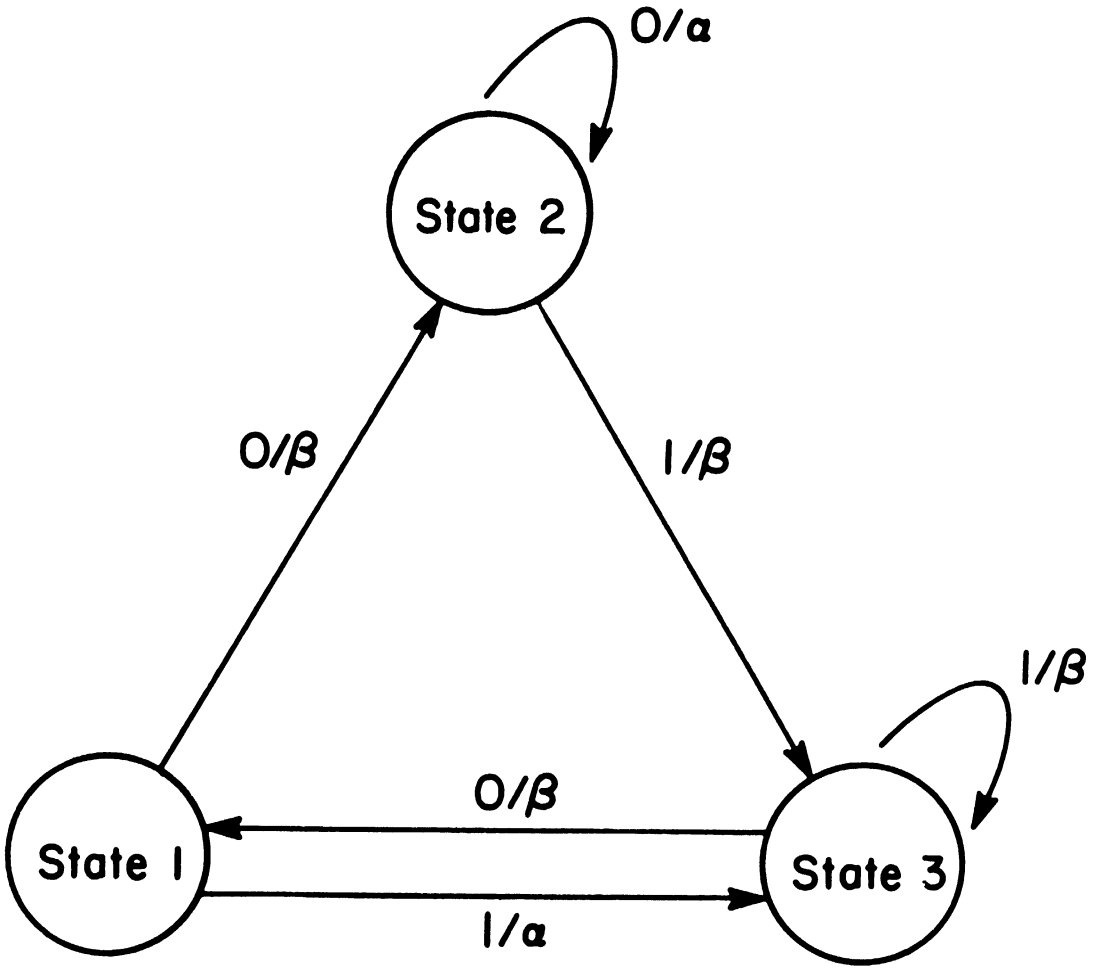


Figure 2.2.3 A finite state machine. The arrow labels indicate the input/output for each transition

which knows the environmental relation in that given an input symbol, the device's output symbol will match the next input symbol. Feedback is obtained by a utility approximation function μ' which at each time step calculates the error in the predictions. Each error can possibly carry a different weight to reflect the magnitude of the error. This process is repeated over several time steps to produce a payoff $\mu(A)$ for each device.

The plan the authors used is likened to evolution: start with some initial randomly generated machine. After obtaining its payoff, generate an offspring machine by randomly "mutating" the "parent" machine. A mutation might be: "add a state, delete a state, randomly change a next state, randomly change the initial state,..." [Fogel,1966] Then the offspring machine is tested and the machine with the best payoff is retained to continue the process. The strategy was varied by varying the number of mutations and putting various biases on what mutations were to be used. In addition, some experiments were run saving up to three parent devices and sometimes producing offspring by combining the parent machines to produce a majority logic machine whose output at each transition is supposed to somewhat reflect the output of all parent machines.

The authors implicitly use the maximum utility criterion and claim to have demonstrated significant adaptation. However, Lindsay in his review of the work has done some rough calculations which indicate that the random search plan would produce performance levels almost as good as those demonstrated by Fogel *et al.* [Lindsay, 1968]. This obviously would not be a random search of the entire space $\mathscr{A}$, but the space investigated by Fogel *et al.* was generally limited to machines having eight states or less.

Lindsay also criticized the fact that the authors do not use most

of the standard concepts of genetics in their use of evolutionary plans. For example, their "organisms" do not at all resemble chromosomes. In fact, when considering their use of finite machines to represent devices, we can see that they run into the same problems evidenced in the other systems we have discussed. In particular, although their mutation operators are not directed towards a particular component of the device *based on a component evaluation function*, their use implies an assumption about the independence of the components (states and transitions) of finite machines. In this context, the independence assumption is extremely bad; furthermore, as Lindsay points out, the representation of a finite machine used by Fogel quite often obscures the structure of the system it is modelling. Fogel's plan is analogous to changing a computer program by randomly adding, deleting or changing instructions. A "learning" system by Friedberg [1958] failed very badly since it tried to operate in this way with similar independence assumptions.

However, even apart from this poor representation, we can pinpoint a bad aspect of the plan. There is generally no provision for storing much information about past experience. A scheme like Samuel's checker player stores such information compactly in the current set of weights, which represent much of the learning that has gone on to date. Fogel *et al's* devices are more specific. The code A is in effect the whole device; Samuel's code is but a mode of operation for a device. Slight changes in Fogel *et al's* devices may produce large changes in observed behavior. This is not the case with Samuel's system. One way to increase the information storage for systems like Fogel *et al's* and Klopf's is to save more devices at each time step. This is in fact the method used by natural evolutionary systems. To summarize Fogel *et al's* system we have:

$\mathscr{A}$ The set of all finite machines. Actually the space used is the the set of all finite machines with less than N states, where N is typically less than 10

$\mathscr{E}$ The set of all possible infinite sequences of symbols

$\mathscr{T}$ The plans which generate finite machines by mutating previously saved machines

μ An overall utility measure based on error counting

χ The maximum utility criterion.

Abstractions and Summary

Now that we have analyzed various adaptive systems within the formal framework we can attempt to compare different adaptive plans looking for common difficulties. In general, we see that with the exception of the evolutionary system all plans considered are continually directed towards reducing some identified deficiency. In GPS and Klopf's system this deficiency is explicitly pointed out using predetermined criteria or evaluation functions. In Samuel's system and Uhr and Vossler's pattern recognition the deficiency is identified as the result of inaccurate prediction. The feedback vector I_t identifies the deficiencies while the transition function $\tilde{\tau}$ attempts to reduce these deficiencies. Moreover, we see that the function $\tilde{\tau}$ is very closely tied to the feedback $\mathscr{I}$, which in turn is very closely tied to the representation of the device. Klopf's plans could possibly be applied to find an optimal detector set for Samuel's system, but only after the problems of identifying bad detectors and generating new detectors were solved. Similarly, GPS could be applied to Klopf's systems, but only after one obtained criteria which identified good and bad aspects of a detector set, rather than an evaluation of each detector function. Fogel *et al's* paradigm could possibly be applied to the other systems but only when each system had a means for accurately evaluating a whole device rather than just components. This last problem might not be as difficult as the others but could involve high time costs in many cases.

We can see from our formal description, however, that many of the systems have common difficulties. The most prevalent of these is the assumption of independence of components. This assumption appears implicitly in adaptive plans which generate a new device by independently

modifying each component of the current devices. This occurs in all the systems we have examined.

Another difficulty can occur when the researcher imparts too much of his own intuition or bias in setting up a representation and *a priori* criteria. The result might be to impose severe restrictions on the set $\mathscr{D}$ or on the adaptive plan. Such limitations will prevent the plan from effectively searching the space $\mathscr{A}$. As Klopf found out, the most intuitive criterion does not always turn out to be the best in practice, especially when additional assumptions are involved. Many plans might benefit from Klopf's method of using a number of criteria before determining which is the best. In other words, one should always try a number of adaptive plans and an evaluation function is a logical element to vary over different plans.

The feedback vector I_t can also be a source of trouble especially when elements like μ' or μ_{c_i} are used. Side stepping the independence problem mentioned above, there is still the chance that the feedback does not really represent what the researcher thinks it represents. Even if the components do not interact, the μ_{c_i} function used may not be the best for ranking possible components or it may contain some inherent statistical error. If the μ_{c_i} functions are to be used to rank the set $\mathscr{A}$, one should check that the induced order is in fact consistent with one's beliefs concerning the goodness of devices. This difficulty is enhanced when the task is not well enough defined. Similarly, care must be taken that enough μ' samples are observed before an estimate of μ is computed and the search continued.

Another difficulty might stem from search techniques. With the exception of Fogel *et al's* scheme, none of the systems discussed saved

previous devices for comparison with the new devices generated. In some cases, this practice might reduce the chance of getting stuck on a local maximum; in other cases it could result in rather erratic search behavior with good devices being discarded. One solution might be to conduct a parallel search, saving a subset of devices at each time step.

Finally, we have seen throughout all the systems studies that the representation of a device and in fact the representation of the entire system plays a crucial role in its success or failure. This is an unfortunate situation for analytical purposes since good representations are often defined with respect to particular tasks. Fogel *et al.* presented what they felt was a good representation in that it was supposed to be generally applicable to various tasks; however, they were not very convincing. We have seen that the comparison of systems with different representations can prove to be very difficult. The formal framework has helped somewhat in extracting various elements of adaptive systems, but the representation dependencies still remain.

Table 2.2 summarizes the difficulties mentioned above. The difficulties are related to $\tilde{\mathscr{A}}$ and $\mathscr{T}$ in our formal framework. Environmental difficulties enter when the environment changes rapidly. This was not the case in the artificial intelligence tasks studies to date. Also, difficulties with a criterion χ were not considered here since many of our examples did not explicitly use a criterion.

Formal Framework Element		Difficulties Involved
$\tilde{\mathscr{A}}$		1) Imposed components may not be independent. 2) Might result in severe restrictions of the set of possible devices. 3) Many other aspects of the system are extremely dependent on representation.
$\mathscr{T}$	$\mathscr{I}$	Feedback difficulties: 1) μ_{c_i} may not reflect true worth of c_i 2) μ may be badly approximated by μ' 3) μ induced by μ_{c_i}'s may not reflect researcher's idea of μ.
	$\tilde{\tau}$	1) Search flexibility might be restricted by *a priori* criteria. 2) May incorrectly assume independence of components.

Table 2.2 Summary of Difficulties

Conclusion

One can draw the following conclusions: first, there are several difficulties which seem to permeate much of the work done in adaptive systems applied to aritifical intelligence problems. This seems to stem from the fact that many of the systems generally use similar techniques. Secondly, it is very difficult to evaluate one system in isolation, yet there seldom exists a comparable alternative system that performs the same task. Furthermore, a very good adaptive system may be concealed under a bad representation, and vice versa.

Where does this leave the adaptive systems researcher? He may continue to attack a variety of tasks using specific representations and tools designed to fit the particular needs of each. He may pay more attention to improving upon certain strategies within a particular $< \tilde{\mathscr{A}}, \mathscr{E}, \chi >$ - system. Or he may try to develop a very general representation and system which could be used for any task.

We have chosen the second alternative with inclinations towards the third. In this way, we can at least make accurate comparisons of plans over the space $\mathscr{T}$ with respect to the criterion χ. Klopf's work serves as a refreshing example of this technique.

The remaining goal of this thesis is to present an adaptive system which will address itself to some of the difficulties mentioned in this chapter. We claim that the system is flexible in its representation and powerful in its searching ability. This system will then be simulated on a computer and attempts will be made to search both the $\mathscr{A}$ and $\mathscr{T}$ spaces.

Chapter 3 Reproductive Strategies

3.1 Search Spaces

As we have seen in the previous chapter, many of the difficulties encountered by adaptive systems are involved in searching a particular space. An adaptive plan is concerned with searching the space of devices $\mathscr{A}$, while the researcher generally has the task of searching the space of plans $\mathscr{T}$. Bagley [1967] has attempted to analyze the difficulty of tasks presented to various adaptive systems by estimating the size of the space involved and also the amount of environmental information which each device must handle. However, there are other aspects of spaces which are probably better indicators of search difficulty than size.

One indicator of search difficulty is the density of points with high payoff. An estimation of this density can be easily obtained using the random search plan. This in fact should be the first step in the study of any system since, as mentioned before, it gives a minimal performance level above which any "good" plan should be able to operate.

Other indicators of search difficulty are probably more significant yet also are more difficult to estimate. Let us review the general search method. The function $\tilde{\tau}$ is responsible for the search in our formal framework. Given an object or set of objects in the space $\mathscr{A}$, $\tilde{\tau}$ generates another set of objects. Typically $\tilde{\tau}$ induces a transformation on the original set of objects so that the derived set bears some resemblance to the original objects. The operation of $\tilde{\tau}$ usually depends on some inherent metric over the space $\mathscr{A}$. For numerical spaces, the euclidean metric is usually used. However, there is not always a suitable metric available for nonnumerical spaces. One can certainly define a metric based upon

how many corresponding elements of two vectors match, but for most purposes this is not adequate.

Discontinuities effect the difficulty of a search space. Continuity is defined in terms of the utility function μ with domain $\mathscr{A}$ and in terms of some metric over $\mathscr{A}$. Numerical spaces like the one used by Samuel are usually continuous. Nonnumerical spaces like Klopf's and that of GPS are likely to have many discontinuities. These discontinuities are often due to the lack of a good metric, but not always. Two detectors may appear very similar (in terms of the functions or computer programs describing them) yet produce quite different behavior when used in a device. This partially explains why the detector problem is more difficult than the weight problem.

Finally, the number of local maxima will increase search difficulty. Multimodal spaces generally are the rule in artificial intelligence tasks, yet no practical search methods presently available can guarantee convergence on the highest peak.

We have seen that the particular representation of a device determines the co-ordinates of the space to be searched, although the underlying task may possibly be represented in a variety of spaces both numerical and nonnumerical. The environment through the utility function μ determines peaks and continuity in the given space. We would like to find an adaptive plan that is capable of operating in a fairly large range of environments and with various representation. Obviously such a plan might not perform better than all plans *in a particular environment*. But since one often does not know what environment is involved, a flexible plan would be a good one to have around. Let us now turn to a possible candidate for such a plan.

3.2 The General Reproductive Paradigm

In this section we will consider a class of adaptive plans which operate in a manner similar to natural genetic systems and in fact use mechanisms which closely resemble genetic operators. However, these plans are not meant to be a model or simulation of evolution. Rather we will try out techniques similar to those which have been observed in natural systems in hopes that they will prove as useful to us in discovering optimal regions in our search space as they have been to the natural evolutionary process.

The general reproductive plan we will be concerned with has been explored theoretically by Holland [1969]. It operates as follows. Initially a set of devices is randomly sampled from the space $\mathcal{A}$. Then the plan operates by producing devices at time t + 1 from the devices at time t in two steps. During step one, called *reproduction*, each device is copied a number of times depending upon how well it performed at time t. The duplication rate for each device is determined by the utility function μ. In step two, called *recombination*, new devices are generated by applying transformation or mixing operators to the set of devices obtained from step one. This procedure is diagrammed in figure 3.2.1.

The reproductive plan is basically a first-order plan (i.e., μ is the only feedback) although it could be modified to use additional feedback as we will explain later. None of the systems we have looked at used first-order plans with the exception of Fogel *et al's*; however, the μ-function did appear explicitly or implicitly in all of them.

Fogel *et al's* plan was based on evolution but they failed to implement the most powerful aspects of reproductive plans. Their popula-

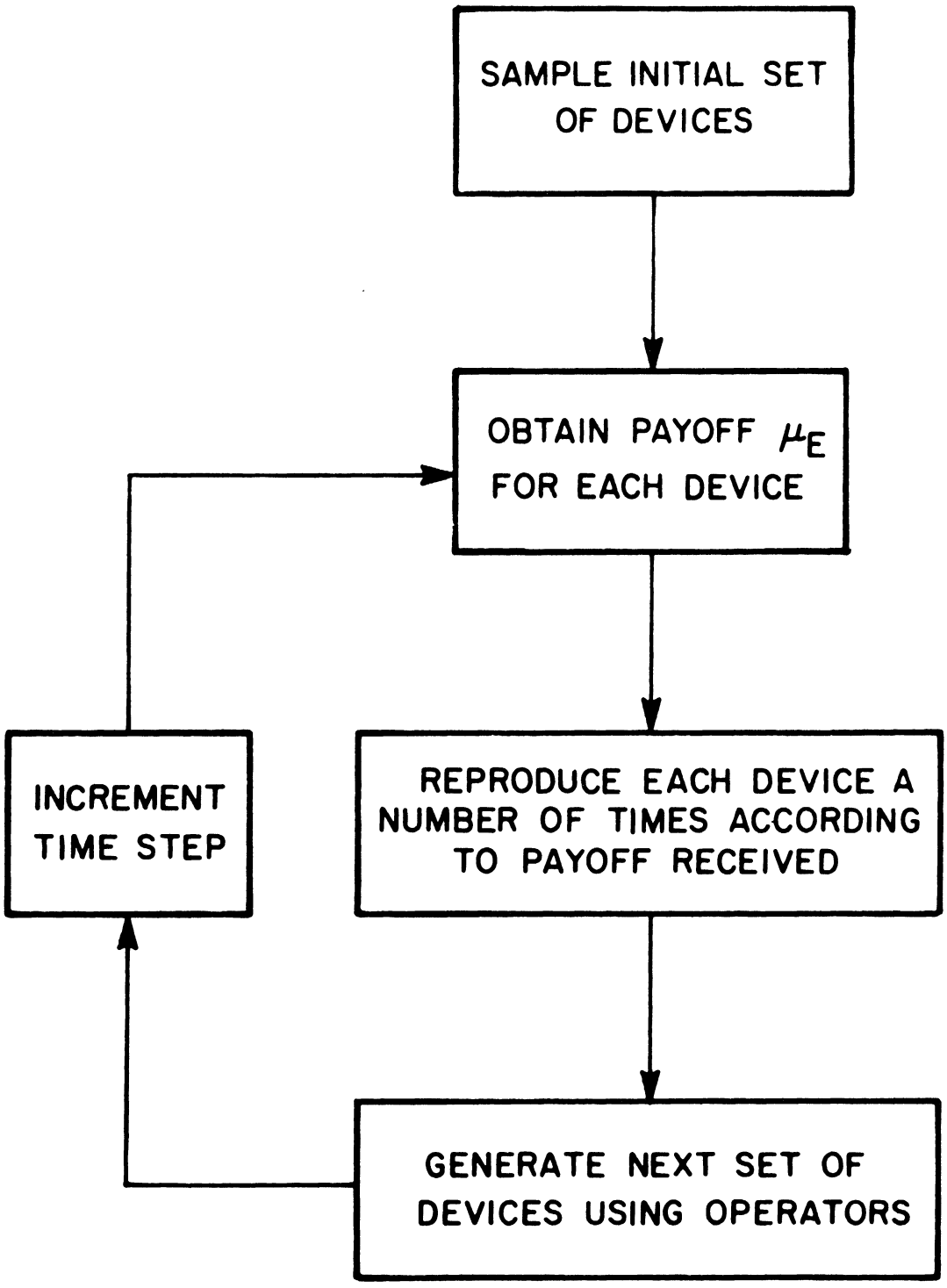


Figure 3.2.1 The general reproductive paradigm

tion usually consisted of only two individuals so that selection amounted to merely picking the best. Larger populations are needed to insure a rich mixture of individual types and to maintain a truly parallel search. Since only one individual produces offspring in Fogel *et al's* system, two very powerful genetic operators (crossover and inversion) were not used. Therefore, their plan was unable to transmit valuable substructures from one device to another.

Holland's theoretical results on reproductive plans indicate that with respect to the accumulated utility function and some given set of mixing operators there exists a reproductive plan which performs at least as well as any other plan in a first-order environment. This result is very important since it at least insures us that we are on the right track in investigating these plans. In addition, Holland's work sets forth the general structure of a reproductive plan. However, it still leaves a wide range of choices when one is concerned with a specific implementation of the plan.

For one thing, Holland's results involve modifying a probability distribution defined over the set $\mathscr{A}$ and computing expected payoffs at each time step. These distributions would have to be approximated since simulations typically do not have the resources to store distribution information about spaces as large as the ones we have considered. Then we must ask how good the approximations are. Furthermore, the process of setting up a suitable set of mixing or transformation operators and the method of deciding exactly how many times to reproduce each individual are not specifically stated in the theorem. Therefore, the space $\mathscr{T}$ of workable well-defined reproductive plans can be rather large.

Bagley [1967] has recently made a good attempt to operationally test

particular reproductive plans using genetic mixing operators as proposed by Holland [forthcoming book]. Performing actual computer simulations he demonstrated that in a game playing situation a reproductive plan could perform as well as a correlation algorithm (similar to that used by Samuel and Klopf), even though the correlation algorithm received more feedback. Furthermore, he demonstrated that as the environment became more nonlinear, the advantages of the reproductive plan became more apparent. Bagley's devices resembled chromosomes and his mixing operators resembled genetic operators (e.g., mutation, crossover). Besides comparing the reproductive plan to the correlation plan, he also investigated various selection schemes (i.e., methods for determining how offspring are distributed) and various genetic operators.

The implementation we will present is similar to Bagley's in its use of operators related to genetics and its basic chromosomal representation. However, it differs from Bagley's in the size and representation of the population, the method of testing and altering the population, the variety of the operators used, and the extent of the subsidiary control mechanisms investigated. Also the space that we will search differs from Bagley's in that it is larger, nonnumerical, and very difficult to analyze theoretically. In one sense the work we will present below is an extension of Bagley's work in that it investigates new mechanisms and extends the search of the space $\mathcal{T}$ of reproductive plans. In another sense it constitutes a different representation of the general reproductive plan and demonstrates its flexibility by applying it to a different, nonnumerical task.

Our implementation of the reproductive plan fits nicely into our formal framework. An element $A \in \mathcal{A}$ will consist of a list or string of well-defined units. The string will be referred to as a *chromosome*

or an *individual* while the units will be referred to as *genes* or *components* in a general sense, or as *subroutines, detectors,* etc. in a functional sense. The notion of a gene as a well-defined functional unit is important here since genetic operators use gene boundaries in their implementation. Although genes may often be further broken down into smaller units, genes will seldom be split at these unit boundaries, whereas chromosomes will often be split at gene boundaries and then reconstructed in various ways. Most of the systems we have looked at employ well-defined functional units which could represent genes in a chromosomal structure. The possible exception is Fogel *et al.'s* system which really needed this type of representation since it purported to model evolutionary mechanisms.

The environment for reproductive plans can be any task which produces a well-defined feedback μ_E for each chromosome taken as a whole. Since this is the only feedback for first-order plans, care should be taken that μ induces as accurate a ranking as possible on the set of devices. Ideally, if two chromosomes differ in only one gene, the function μ should indicate this difference to the extent that it affects the device's performance. If these different genes do not affect performance, then the chromosomes should be rated equal.

Now let us consider more specifically the composition of the space $\mathscr{T}$ and the methods which will be used to search it. Figure 3.2.2 presents a flow chart outline of the paradigm. This figure should help the reader follow the logic. A few definitions will be helpful here. A *generation* will constitute the basic time unit in the operation of a particular reproductive plan. Each generation involves a sampling of a set of devices in the space $\mathscr{A}$. A *run* will consist of the entire process of sampling successive devices by a particular

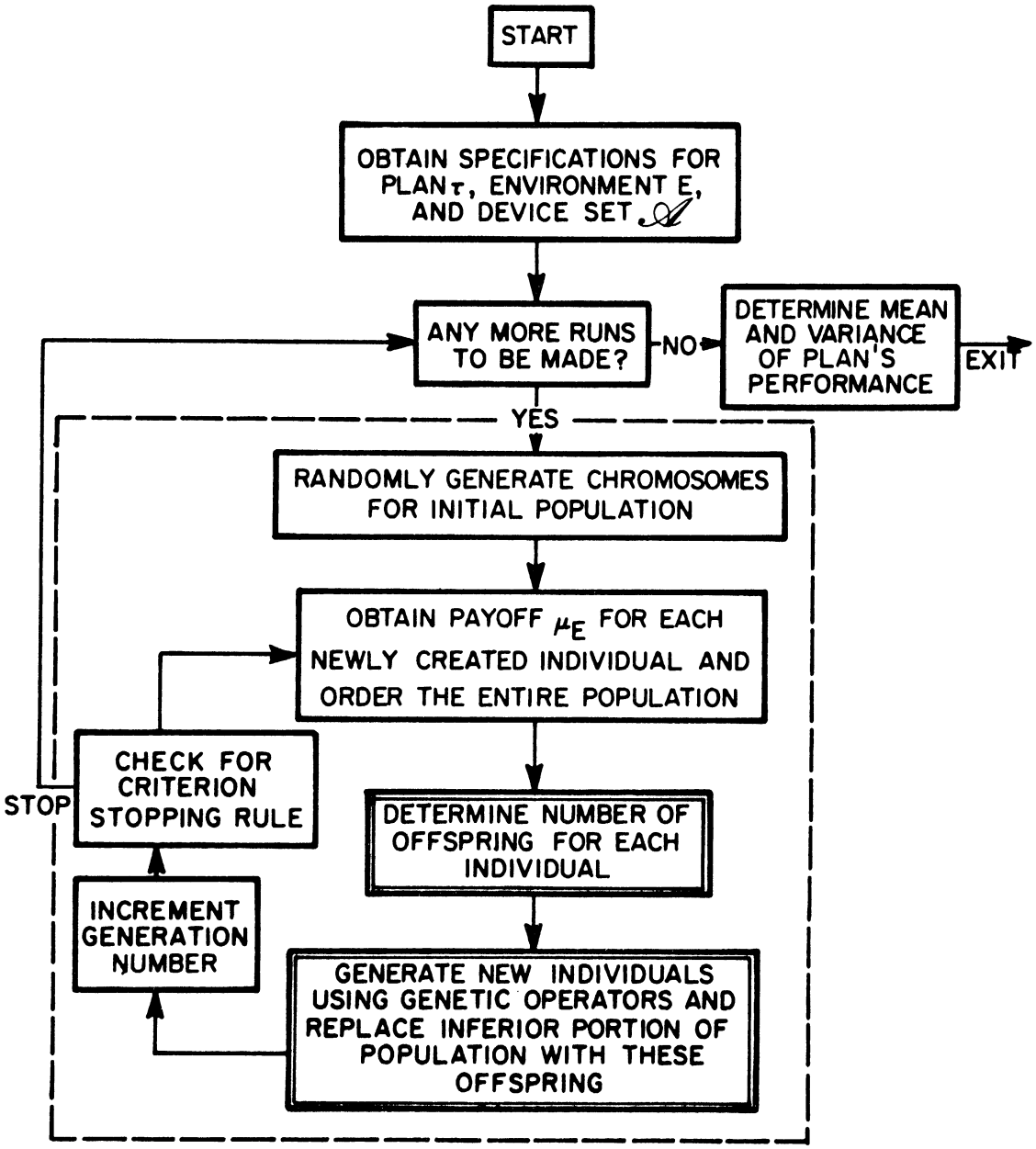


Figure 3.2.2 Outline for testing reproductive plans

plan until a certain stopping rule is exercised. This rule might be determined by the $f(\tau)$ function used by the maximum utility criterion discussed above. At this point in the investigation, the performance of the best individual (or set of individuals) in the run and the number of generations used are saved as an estimate of the plan's goodness under some criterion. This is only an estimate since the reproductive plan contains random processes. Therefore, the goodness of a plan can only be described with a distribution of final outcomes or alternatively some parameters of this distribution such as the mean and variance. A *test* or *experiment* will refer to the process of operating a number of runs in order to estimate the mean and variance. In figure 3.2.2 the process of one *run* is enclosed in a dotted box.

The first operation in Figure 3.2.2 is the specification of the set $\mathscr{A}$, the environment E, and the particular plan τ. The specification of $\mathscr{A}$ will consist of setting a maximum length for each chromosome in $\mathscr{A}$ while E will specifically indicate the task to be carried out. The specification of $\mathscr{A}$ and E will usually remain the same over a number of experiments so that various plans can be accurately compared. The specification of a plan τ will involve stipulating a number of parameters such as population size, percent of the population to be replaced each generation, and probabilities of applying certain genetic operators. This specification may also indicate the use of different schemes at various point in the plan. The operations in Figure 3.2.2 which are enclosed by a double set of lines are operations for which a number of different schemes will be tested. Also additional controls will be introduced which generally will increase the complexity of the diagram. One could imagine all such schemes and controls existing sim-

ultaneously in the general plan, with a set of switches to determine which schemes would be used for each specific plan. However, to keep the exposition as clear as possible the flow chart will be modified only with respect to the operation under consideration. If a certain scheme proves to be superior to others, then it will assume exclusive responsibility for that operation. However, if a number of schemes work adequately, then they will be retained as conditional operations in the flow chart. In this way we will be able to follow the search of the $\mathscr{T}$ space by following the modification of the flow chart.

A particular run proceeds as follows. The initial population is randomly sampled from the set $\mathscr{A}$. Next each individual is ranked according to how well it performed the particular task. Since we will have an accurate performance measure μ_E, rather than an estimator μ', we need obtain payoff for each individual only once. The top portion of the ranking is designated as the *current population* and each population member is assigned an integer number (again according to μ) to indicate the extent to which it will be used in producing offspring. Those individuals with nonzero numbers constitute the domain of the genetic operators. The genetic operators generate a fixed number of new individuals each generation by "mixing" duplicates of other individuals and adding a certain amount of random "genetic material". This process will be discussed in more detail below. Then the new individuals "compete" for positions in the current population with the best possibly replacing some previous population members. After each generation a check is made for a stopping condition dependent upon what criterion is being used.

Let us consider in a little more detail the question of a criterion

for ranking reproductive plans. Intuitively such a plan should be both efficient and effective. An *effective* plan should be able to discover substantial peaks in the space $\mathcal{A}$. In addition an *efficient* plan should be able to find these peaks with a minimum amount of sampling. These two criteria, however, cannot always be considered independently and therein lies a problem. If one has knowledge of the maximum possible performance level and he feels that this level can be obtained without extreme effort (or alternatively that this level must be obtained regardless of the effort) then the effectiveness criterion is well-defined by this performance level. Therefore, efficiency becomes the only real question and we can use the *minimum time criterion.* If μ^* indicates the maximum possible performance level, then:

$$\chi(\tau) = \min \left\{ t \mid \mu(A_t^\tau) = \mu^* \right\}$$

When we wish to compare plans which sample a different number of devices each time step we will use a more basic criterion, the *minimum sample criterion,* which measures the total number of *samples* before an individual with payoff μ^* is obtained. These two criteria are equivalent with respect to plans which sample the same fixed number of devices each time step.

When it is the case that only a specified number of samples is allowed for each plan, then effectiveness is of primary concern. In this case, the maximum utility criterion would be used with $f(\tau)$ constant.

Unfortunately for many tasks we are not sure what level of performance is reasonably attainable nor are we willing to rigidly specify a maximum amount of time we will let the plan operate. What type of a stopping rule would be reasonable in this situation? One possibility

is a marginal utility rule. In other words, we would let a plan continue as long as the performance has improved at least a certain amount in the last n time steps. This stopping rule would effectively define $f(\tau)$ in the maximum utility criterion. This use of the maximum utility criterion could be criticized on the grounds that comparisons are made between plans obtaining an unequal number of samples. However, there does not seem to be a viable alternative without *a priori* knowledge of the search space. Therefore, this criterion will be used for most of the experiments that we will perform. However, when there seems to be a significant difference between the average numbers of samples used by alternative plans, this difference will be considered in the final analysis. This difference will be particularly important when the average performance levels do not differ significantly.

Let us now look at the genetic operators in more detail. The following distinctions will be useful when considering certain chromosomal representations. A *gene* will be thought of as a functional entity which exerts control over certain processes in the device. An *allele* will be a possible substitution instance for a gene which will specifically pinpoint how the function is to be performed. For example, if Samuel's weights were interpreted in this structure, the functional notion of a gene would be associated with each detector, while the alleles for each gene would be all possible weights which could be attached to each particular detector. However, in Klopf's system this distinction is not necessary since all detectors play the same role.

We will now define three genetic-like operations which are generally applicable to any chromosomal representation regardless of the task. We should mention that these operations and the assumptions involved in their use do not necessarily correspond exactly to their counterparts

in living systems. However, we will try to maintain all the evolutionary aspects of living systems which we feel will benefit our adaptive plan within our space and time limitations.

Let us use Roman letters to represent genes or particular alleles in a chromosome, and arrows to designate breakage points. The *crossover* operator acts on a pair of strings by breaking each string at some point and rejoining the subsegments from different strings. Figure 3.2.3 illustrates this process:

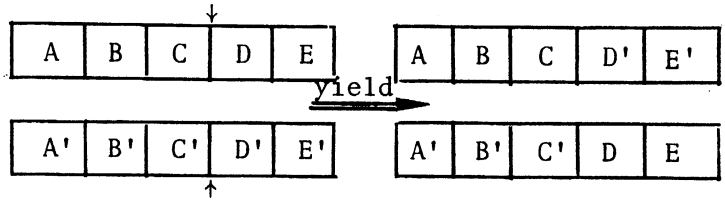


Figure 3.2.3 The Crossover Operator

For our purposes breakage will be equiprobable at every point, although the breakage points could be determined according to some nonuniform distribution. Also certain conditions might have to be satisfied before crossover may take place, such as a similarity condition between the two input strings. However, we will just pair strings randomly. The term *double crossover* will refer to the crossover operator with breakage at two points in each string, resulting in the exchange of inner substrings:

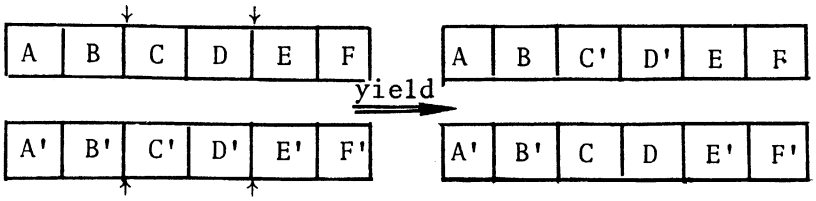


Figure 3.2.4 Double Crossover

The *inversion* operator also makes two breaks, inverts the inner segment, and then rejoins the string:

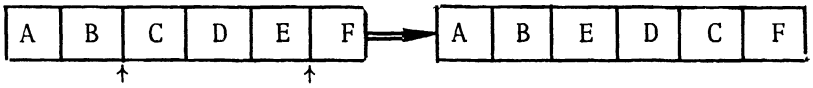


Figure 3.2.5 Inversion

Inversion may or may not change the performance of a device, dependent upon the existence of order dependencies between genes. However, in conjunction with the crossover operator it can bring about important changes in adaptation as we will discuss below.

The role of crossover and inversion as valuable mixing operators has been pointed out by Holland [forthcoming book]. Let us consider a few important points here. Crossover (and double crossover in the following discussion) is capable of sampling a wide variety of devices in the space while still preserving certain nonlinear interactions that might exist between genes. It enables a plan to take large steps in the search space as opposed to small, local steps. Therefore, even if an offspring resulting from crossover obtains a payoff which is only comparable to that of its parents, there is still a net gain since now a new region of the search space is being investigated and the parallelism of the plan is extended. However, the offspring is often much better than both parents due to the combination of subsegments which work very well together. Due to the variability in breakage points and the use of two types of crossover, the subsegments exchanged vary widely in length and composition, enabling the plan to test a large combination of devices.

Inversion plays a dual role. On the one hand it sets the stage so that later mixing by crossover is beneficial. Quite often the situation may arise in which a number of individuals in the current population differ only at a few genes. Crossover using these individuals will most likely result in little or no significant mixing. Inversion changes this situation without necessarily changing the devices involved. In this way it enriches the set of possible future devices to be sampled.

Inversion also plays another role in co-ordination with crossover in nonlinear situations. Consider the alleles P and P' of two different genes. If P and P' occur on the same string and are positioned very close together, then we will say they are highly *linked*. The farther apart, the less they are linked. Let us assume that interactions in the use of P and P' are such that when they occur together on one chromosome, the performance of the resultant device is much higher than would be warranted considering either allele by itself. In other words, we have an extremely nonlinear situation. These two alleles could be assembled on one string with the crossover operator. However, the chances that they are highly linked on the resultant string are rather low. Therefore, the chance that they will again be separated by another crossover operation are rather high. Inversion obviously modifies linkage. If P and P' become highly linked via inversion the chances that they will be separated becomes very low. In addition, the string containing P and P' will produce a large number of offspring due to its high performance. Since most offspring will contain this highly linked P P' unit, the chances that this unit will be lost to the population are extremely low, even if P and P' are separated in a few individuals.

The third type of operator we will be concerned with is mutation.

As opposed to crossover and inversion, mutation will be specifically designed according to the structure of the chromosomes. Since the basic purpose of mutation is to introduce new alleles, the mutation operator must have knowledge about what constitutes a permissable allele. A *complete* mutation will result in the whole allele being replaced. If some substructure is present in an allele, a *partial* mutation may replace only a portion of the allele. A mutation is *random* if the new allele or portion is randomly taken from all possible alleles. A mutation is *biased* if the selection process is biased, perhaps by some metric defined over the alleles or allele substructures. We will generalize the notion of mutation so that it may map a set of alleles into another set. Operating on the empty set we will call the operator *insertion*. If the resultant set is empty we will call it *deletion*. These will be considered special cases of mutation.

In the systems analyzed above, we have seen examples of mutation-type operators. Klopf's scheme replaced certain inferior detectors with randomly chosen ones. Fogel *et al.* used insertion and deletion in addition to random mutation. He also changed his initial state in a non-random manner. Insertion, deletion and random mutation are generally applicable to any representation as long as the set of admissable alleles are known. However, the researcher can often cut down the range of the mutation operator so that the space can be searched more locally. Fogel *et al.* changed the initial state designation for a particular machine to the next state in the transition in an attempt to check for errors due to phase shifting. Similarly, Klopf could have made smaller changes in his detectors either by changing the detector output for only a few of the input patterns, or by incrementing or decrementing the values rather

than using new random values. Since one purpose of the mutation operator is to perform local searches, it should be capable of taking as small a step as possible at the intragene level. This proves to be very useful in the later phases of adaptation when local steps are more important and larger steps are often harmful.

Now we have seen how crossover, inversion, and mutation play important complementary roles in searching the space $\mathcal{A}$. Along with the selection and reproduction phases, they direct the search of $\mathcal{A}$ by progressively skewing the distribution of sampled devices towards devices having higher and higher payoff. The initial sample is random and therefore uniform over the space $\mathcal{A}$. Selection eliminates the worst devices while reproduction skews the distribution towards the better devices. Then the genetic operators sample new devices by recombining various aspects of old devices and adding a bit of new genetic material through mutation. This process usually results in the retention of at least a few new and better devices which further skew the distribution. Thus we see all aspects of the reproductive plan interacting to achieve adaptation.

Now we must consider what are the important aspects of reproductive plans and how these aspects should be investigated. One important aspect is the population size. The population contains most of the information the plan has gathered to date. Furthermore, the operations of crossover and inversion are only successful when there is sufficient variability in the population. So in addition to being large, the population must have a number of *different* individuals.

Population size is one aspect which will be investigated experimentally. However, the maximum size will be put at about 40 individuals.

A minor reason for this is space limitation. More important, however, are time or cost considerations. Since the cost per sample is directly measured in the criterion, we must be very careful to sample as little as possible while still preserving the power of the plan. Since the number of samples per generation will range only between 2 and 10, it seems that a population of size 40 or less will provide sufficient diversity with proper care.

Proper care in one respect amounts to investigating various selection schemes. A selection scheme is a method of apportioning offspring to members of the current population. At one extreme a selection scheme could emphasize the best individuals to such an extent that the resultant populations become very homogeneous, each individual closely resembling one of a limited number of parents. Such a situation is undesirable and effectively lowers the population size. On the other hand, insufficient selection will slow down the skewing process and often result in inefficient adaptation. Since the population sizes are rather small and the number of samples per generation even smaller, we must be very careful that offspring are distributed in as optimal a method as possible.

We will also study the effects of the genetic operators. In particular, we would like to know how often and how many times each operator should be applied. Too much mixing may result in only large steps being taken in the space while too little mixing may decrease the variability in the population. Since the operators are to be applied probabilistically, this optimization could amount to parameter setting. However, the need for various operators may change during evolution or over different tasks. This would call for more sophisticated controls.

Our paradigm may also be improved by introducing additional controls and applying other aspects of natural evolution. Dominance provides for variability with reduced selective pressure. Intrachromosomal duplication locally increases the sampling rate for a particular gene. Isolated populations interbreed with a resultant increase in variability. These and similar techniques will be investigated to see what effect they may have on adaptation. Hopefully, the more sophisticated the plans get, the more information we can extract about the operation of reproductive plans so that the search process will become more refined. This will not involve receiving more information from the environment but will involve inferring more information about the search space by observing techniques which work and testing for situations in which they have been shown to work. In addition, there will be a constant effort to develop techniques which could prove to be generally applicable rather than specific to a particular task.

Summary

In this section we have presented a paradigm for an adaptive plan which we claim should overcome some of the difficulties encountered by other plans. Of most importance is the ability of reproductive plans to deal with nonlinear interactions between subunits or components of the device. This is accomplished using the crossover and inversion operators. The population which enables the plan to use crossover and inversion effectively also enables it to conduct a parallel search. This is another important aspect since it eases the problems involved in searching multimodal spaces. In addition, the plan is first-order and therefore does not restrict itself by making assumptions about the nature of other kinds of feedback. Furthermore, the performance of a good first-order plan sets a nontrivial lower bound on the performance of plans which receive additional feedback.

In our last chapter we will speculate as to how additional feedback could possibly be used by a reproductive plan. The immediate goal now, however, is to find a suitable task with which to study our reproductive plans.

Chapter 4 Task Selection and Initial Attempts

4.1 The Question of Cost

It will take various amounts of computer time to perform different tasks. In addition, different plans may require different amounts of time on the same task depending upon how much information is extracted and processed by the plan. In a modelling situation real time computer costs do not directly affect the way in which the investigator sets up his model. Certainly he must live within the space limitations imposed by his budget, but these limitations do not typically enter into his criterion for the model's success.

On the other hand, the process of searching the $\mathscr{A}$ space associated with an artificial intelligence task usually involves higher time costs than does a typical simulation of a model. Also there is a hope that good adaptive plans can be generalized so that they can be used many times for production runs. In addition, some tasks require adaptive systems to operate in real time. Typical optimization techniques are good examples: a control system for an automatic pilot or a robot exploring a far off planet are systems which might need a real time adaptive plan. So we see that real time cost is an important aspect which can play a major role in the development of an adaptive system.

Uhr and Vossler in their work with an adaptive pattern recognition scheme realized this cost problem [1963]. They developed an elaborate system which extracted many bits of information from patterns, only to discard the system for one which operated more quickly. The original system took about 40 seconds to process one character while the modified system reduced this to one second. Certainly time sta-

tistics like these are dependent upon particular computers, yet computer differences do not usually account for time factors of 40 or more.

The biggest costs for adaptive plans usually consist of the cost of obtaining feedback or, in other words, the cost of sampling new devices in the space $\mathscr{A}$. In general, the more feedback received, the greater the cost. Let us consider first-order plans where the only feedback is μ_E for each device. How can we classify the cost of obtaining μ_E as being high, medium, or low? If we were working with a real time system we would have to operate within a fixed limit; the plan would have to produce a new device at least as soon as it was needed. A high cost situation might be one in which only one or two devices could be tested before the plan was required to submit a new device for actual use. When real time operation is not important, the judgement of the cost might depend on how long the researcher wants to wait for the plan to terminate or how big his budget is.

A more consistent method of evaluating cost is to compare the time needed to obtain the feedback μ with the time needed to process and analyze this information and generate or select another sample. For reproductive plans most of this processing takes place in producing new offspring. The time to complete this process is less variable over different tasks than the time required to obtain the feedback. If these two times are comparable within an arbitrary factor then we might say the cost for this task is normal. This cost will play an important role in the operation of our plan. In a high cost situation we will judiciously take only a few samples per generation. In a low cost situation we can maintain a large population and obtain many samples. This cost will directly affect our investigation. Ultimately, however, we should use the minimum

sample criterion. Then time cost is not the only concern and opportunity costs become relevant. In other words, how well can the plan do with what information is available?

4.2 Statistical Tests and Some Comments on Variance

As mentioned previously, each experiment will be run a number of times, thereby generating a distribution of final performance values. Then according to the criterion currently being used, the distributions and, therefore, the plans which generated the distributions will be ranked.

Since it would be impractical to experiment with all possible reproductive plans, we will not be concerned with finding a plan which is optimal with respect to the given criterion. Rather we will be concerned with generating successive reproductive plans which are as good as or better than previously tried plans.

A suitable statistic for ranking different adaptive plans is the mean of the experimental distribution. Depending on the criterion used, this statistic may be the mean number of samples required to achieve a certain performance level, the mean performance level after a fixed number of samples, etc. A suitable statistical test to determine significance levels between plans is the difference-in-means test.

The standard "difference-of-two-means test" assumes that the underlying distributors are normal [Hoel, p. 276]. As we will see later, the distribution of the random sampling plan closely resembles the normal distribution. Also since the final results of each run are influenced by a number of random processes, one can cite the central limit theorem to argue that these results will be normally distributed. For

our purposes we will use Student's t distribution to eliminate the error in approximating the normal distribution with small sample sizes. This is very important since we generally will run each experiment only five times, due to the relatively high cost involved.

Another question comes up with regard to the alternative hypothesis used in the statistical tests. Let us designate $\bar{\mu}_x$ and $\bar{\mu}_y$ as the distribution means for plans x and y. The statistical hypothesis H_o will always be:

$$H_o: \bar{\mu}_x = \bar{\mu}_y$$

The alternative hypothesis may be:

$$H_1: \bar{\mu}_x > \bar{\mu} \quad \text{or}$$

$$H_1': \bar{\mu}_x \neq \bar{\mu}_y$$

depending upon our *a priori* feelings about plan x and plan y. For most of the tests we will use H_1 since we generally will have reason to believe that each new plan we develop will perform better than the previously best plan. However, sometimes we will be adjusting parameter values without any *a priori* feeling for the best value. In these cases we will use H_1'. We will use the standard 5% level of significance for all of our tests. When results are much more significant than the 5% level, we will mention the significance level.

It is often difficult for one to estimate the significance of the difference of two means by inspecting only the means. The variance of the distributions is also important; the larger the variance the less significant the difference. The reader should bear this in mind when reviewing our results.

Sometimes the sample variances of two distributions differ by a very large amount. Therefore, we will also test to see if the variances

are significantly different [Hoel, p. 225]. If this is the case we will have to use a different test (which does not assume equal variances) to determine if the difference-in-means is significant [Hoel, p. 279].

However, the variance by itself is an important statistic for evaluating adaptive plans. A plan that induces a small variance over various runs is better than one with a large variance, given that the means of the two plans are not significantly different. A small variance is important since in production situations one usually will only make one run. In this case we would like assurance that this run is fairly representative of the distribution. One phase of our investigation will be concerned with forcing all runs of a plan to reach a comparable level of performance, thereby decreasing the variance between runs.

A clarifying word on terminology is needed at this point. Throughout our investigations we will use the word *variance* in two different senses. This should cause no confusion as long as the reader is aware of the distinction. In one sense, we will speak of *variance between runs of an experiment*. This is the variance mentioned in the above discussion and refers to the spread in performance values over different tests of our adaptive plans using different random numbers. We would like this variance to be low, indicating that all runs converge to comparable performance levels.

For our second use of the word we will usually speak of *genetic or population variance*. This variance is an indication of the variety of individuals in the current population at a specific point in time. As opposed to the variance between runs, we would like the population variance to remain high throughout evolution. A large population variance is necessary in order to maintain a truly parallel search. If many pop-

ulation members resemble one another, we find ourselves searching only a small area of the space $\mathcal{A}$ since current population members determine where new samples will be taken. Furthermore, Crow and Kimura [1970] have shown theoretically that under the conditions of a fixed environment and random mating, the rate of increase of the average performance of a population is approximately equal to the genetic or population variance suitably defined. Later in our investigations we will experiment with techniques which attempt to improve the performance of reproductive plans by continuously insuring that population variance is maintained.

4.3 Initial Attempts

The problem now at hand is to find a suitable task for experimenting with reproductive plans. Initially we felt it best to operate reproductive plans on a low cost task so that we could get a feel for the operation of these plans and gain some insight into possible improvements on the basic reproductive paradigm.

The task we have used is somewhat artificial although it includes features that are common to typical artificial intelligence tasks. The task may be defined as follows: We have available a set of 50 subroutines. Specific subsets of these subroutines are needed to perform certain macro operations. An optimal device must contain all necessary subroutines to completely perform a number of macro operations and furthermore must perform these operations in a minimal amount of time.

This type of task could arise in a number of situations. For example, the task might be to build the necessary parts for a machine. A macro operation could constitute the construction of a part in an assembly line operation, while the subroutines might constitute the primitive steps required to put the part together. There might be a number of ways in which a part could be assembled, each varying in cost. Some operations

may be dependent upon others. In a pattern recognition situation the subroutines might correspond to detectors while the macro operations could correspond to the successful recognition of a pattern.

We will assume that there are no order dependencies between the subroutines or macro operations. The task is represented in a chromosome as follows. A gene consists of a single subroutine. The chromosome will contain at least enough gene positions to perform all necessary macro operations, given that the necessary subroutines are available. The task is performed in a serial manner by operating the subroutines from left to right on the chromosome. Each subroutine takes one time unit to run. When all the subroutines for a *particular macro operation* have been performed, the total elapsed time is recorded. When all necessary macro operations have been completed, processing is terminated. Of course, the chromosome may be exhausted before all operations have been completed, in which case partial credit is assigned along with the maximum operating time. To be more specific, payoff μ is calculated in the following manner. Let

$$\mu_i = \frac{S_i}{S_{i_{max}}} + \frac{T_{max}-T_i}{T_{max}}$$

where μ_i = the payoff of the ith macro operation

$S_{i_{max}}$ = the total number of subroutines needed to perform the macro operation i

S_i = the actual number of operation i subroutines found in the chromosome

T_{max} = the length of the chromosome

T_i = the amount of time required to reach and perform all of operation i's subroutines.

If all of a certain macro operation's subroutines are not present, T_i would equal T_{max}. Payoff μ is equal to the average of the μ_i's over all operations needed to accomplish the task.

The reproductive plan receives only the payoff for each string. Crossover and inversion and a single random mutation are applied to each chromosome pair with certain probabilities. Two special operators are also used. A *deletion-insertion* operator extracts a subroutine from some random position on the chromosome and inserts it at another random position. The inner subroutines are shifted one position in the process. A *reversal* operator exchanges the position of two randomly chosen subroutines.

As one can surmise, the special operators were designed for this specific task. However, their use does not presuppose detailed knowledge of how the payoff is obtained. All we assume is that the position of a subroutine in the chromosome will have an effect on performance. Certainly inversion also changes positions but in a grosser manner. Mutation is the only operator that can introduce new subroutines into the population, although crossover can assemble new subroutines on one string if they are already present in the population. Mutation must be completely random since there is no similarity measure between subroutines. We are operating in a nonnumerical space with no particular order relation.

We will use this task to investigate the effects on performance of different operator probabilities and different selection schemes. Operator probabilities are important since they determine to what extent duplicates of population members will be mixed and mutated to form new individuals. Too much mixing may be detrimental while not enough may result in an inefficient search. *Selection schemes* are methods based upon individuals' performance of determining to what extent population members will

be duplicated to serve as input to the genetic operators. Overselection, a situation arising from duplicating a few very good individuals too many times, may result in a loss of population variance. However, underselection may not influence sampling enough to induce an efficient search.

Two selection schemes were used in our experiments. With Selection Scheme 1, a certain percentage of the top individuals is chosen to serve as parents. The rest of the population will be replaced by new individuals. The payoffs of this group are then rescaled by subtracting the lowest payoff of the group from all others. The resultant values are used as weights to determine how many offspring each indiviual will contribute. With Selection Scheme 2, the offspring are equally distributed as far as possible among the parent portion of the population.

The task used for the initial experiments involved three macro operations, each of which required five subroutines. Some subroutines were used by more than one macro operation. The first set of experiments were designed to test the plan's sensitivity to the genetic operator probabilities. The following parameter settings were used:

Population size ----------25 individuals

Percent of population serving as parents --30%

Chromosome size ----------25 subroutines

Selection Scheme ----------1

Only 12 subroutines were needed to complete all macro operations due to overlapping use.

We observed that most of the first few runs reached the maximum payoff after an average of about 80 generations (1360 samples). An individual receiving the maximum payoff contained all the needed subroutines in the 12 leftmost positions of the string, arranged so that certain macro operations were completed after only five or six time steps, thereby minimiz-

ing the average completion time. Since the maximum utility was rather easily obtainable, the maximum utility criterion (after a fixed number of generations) did not seem suitable. Therefore, the minimum sample criterion was used (or equivalently in our case the minimum generation criterion). Also, instead of operating the plan until a maximum utility individual was discovered, runs were terminated when an individual was discovered with utility equal to 98% of the maximum. This seemed reasonable since some runs would get stuck on minor suboptimal peaks at the level of about 99% of maximum utility. All runs were terminated at or before the 150th generation whether or not the desired payoff level had been obtained.

Tables 4.3.1, 4.3.2, and 4.3.3 give results for the more significant experiments performed. The meaning of the column headings is as follows:

EX NO ----- An identification number for each experiment

SEL SCM --- The selection scheme used

MUT ------- Mutation probability

D-I ------- Deletion-Insertion probability

REV ------- Reversal probability

INV ------- Inversion probability

CRS ------- Crossover probability

% MAX ----- The percent of the *runs* in this experiment which reached the maximum allowable number of generations (150), before attaining the desired payoff level.

MEAN ------ The mean number of generations used by the runs in this experiment. If a run reached the maximum allowable generation (150), then the value 150 was used in the computation of the mean.

S.D. ------ The standard deviation of the runs in this experiment. If any runs reached the maximum allowable generation, the standard deviation is not listed.

SIGN ------ Indication of which experiments were significantly different from others using the minimum generation criterion. These experiments will be referred to in the text.

The first set of experiments was designed to test the plan's sensitivity to operator probability levels. All operators were applied with the same probability. Table 4.3.1 summarizes the results of these experiments. Only very high and very low probability levels resulted in significantly inferior performance. Experiment 1 with low probabilities was the worst. This could be attributable to two factors. First, there was a high probability (greater than .5) that an offspring would be affected by none of the operators and therefore obtain the same payoff as its parent. Since this situation was not checked for, this "new" individual would then enter the parent portion of the population and effectively decrease the population variance since its parent also remains. The second reason for this experiment's inferior performance, is the low mutation rate. This is discussed below. Experiment 5, using very high probability levels, performed better than experiment 1, but still differed significantly from the experiments using intermediate levels. The inability to take small steps in the search space probably accounts for the inferior behavior of this experiment.

EX NO	SEL SCM	OPERATOR PROBABILITIES MUT	D.I.	REV	INV	CRS	% MAX	MEAN	S.D.	SIGN
1	1	.10	.10	.10	.10	.10	10	112.1	-	*
2	1	.30	.30	.30	.30	.30		50.3	11.0	
3	1	.50	.50	.50	.50	.50		51.1	11.5	
4	1	.75	.75	.75	.75	.75		47.7	17.0	
5	1	.99	.99	.99	.99	.99		61.7	7.2	*

Table 4.3.1 Probability Levels

The next two sets of experiments were designed to test the plan's sensitivity to individual operators. We consider mutation a very important operator for this task since it is the only operator which provides subroutines not already existent in the current population. The experiments in Table 4.3.2 demonstrate that a 20% mutation rate is sufficient to enable the plan to operate as efficiently as the plans in experiments 2-4. Lower mutation rates were very detrimental even with intermediate probability settings for the other operators. Many of the runs in these experiments failed to reach the desired utility level in the allowable number of generations.

EX NO	SEL SCM	OPERATOR PROBABILITIES MUT	D.I.	REV	INV	CRS	% MAX	MEAN	S.D.	SIGN
6	1	.10	.99	.99	.99	.99	40	115.1	-	
7	1	.20	.50	.50	.50	.50		53.4	17.7	*
8	1	.10	.50	.50	.50	.50	20	109.1	-	
9	1	.00	.50	.50	.50	.50	100	150.0	-	*

Table 4.3.2 Mutation Levels

EX NO	SEL SCM	OPERATOR PROBABILITIES MUT	D.I.	REV	INV	CRS	% MAX	MEAN	S.D.	SIGN
10	1	.50	.00	.00	.00	.50	90	148.5	-	*
11	1	.50	.10	.10	.10	.50	10	76.5	-	
12	1	.50	.00	.00	.50	.50	20	105.4	-	
13	1	.99	.00	.00	.99	.99		71.3	20.0	*

Table 4.3.3 Order-Shifting Mechanisms

Mutation is important in obtaining the necessary subroutines to complete all the macro operations. Deletion-insertion, reversal, and inversion are capable of moving these subroutines around to minimize processing time, the other factor used to determine the payoff. Table 4.3.3 shows the effects of these operators. When all of these order-shifting operator probabilities were set to zero, performance was very poor (EX.NO. 10). A slight increase in these probabilities produced a significant improvement (EX.NO. 11). Inversion without reversal and deletion-insertion was also helpful. However, inversion provides for grosser position shifts than the other two operators. This is not desirable in the later stages of adaptation.

One aspect of these experiments became very obvious upon closer examination of the runs. Those runs with high mutation rates did very well initially. However, after all necessary subroutines had been obtained, a high mutation rate could be detrimental by eliminating some of the necessary subroutines. At this point in adaptation a low mutation rate and high rates for order-shifting operators would be advantageous. Experiments whose probability settings satisfied this latter condition (e.g., EX.NO. 7) performed well once all necessary detectors had been discovered.

Other experiments were run with an assortment of probability settings. None of these performed significantly different from experiments 2, 3, 4 and 7. We may therefore draw the following conclusions:

1) There is a general insensitivity to operator probability settings, except in the extreme cases. As long as mutation and some position shifting operator were relatively active, performance due to different probability settings did not vary significantly.
2) There seemed to be a need to have operator probabilities vary *during* adaptation.

Figure 4.3.1 shows the performance of some of the experiments from Table 4.3.1. The average performance of all runs at each generation is plotted. Figure 4.3.2 shows some sample runs for experiment 4. This indicates the variance involved between runs. The highest payoff observed after each generation is plotted.

The next set of experiments tested the effect of the selection scheme used. Two probability settings which were "good" using Selection Scheme 1 and one "bad" setting were rerun using Selection Scheme 2. Table 4.3.4 shows that the "good" settings turned in significantly poorer performance (EX.NO. 14 vs. EX.NO. 3, NO. 15 vs NO. 7) using scheme 2. Therefore, we may conclude that the weighted selection scheme results in a superior plan.

A few more experiments were run to test the effect of varying other parameters of the plan such as population size, percent of the population retained after each generation, and the size of the chromosome. Of most importance was chromosome size. When the number of permissable subroutines on the chromosome were reduced to 20 and then to 15, performance decreased markedly. This may be directly attributable to the lack of redundancy

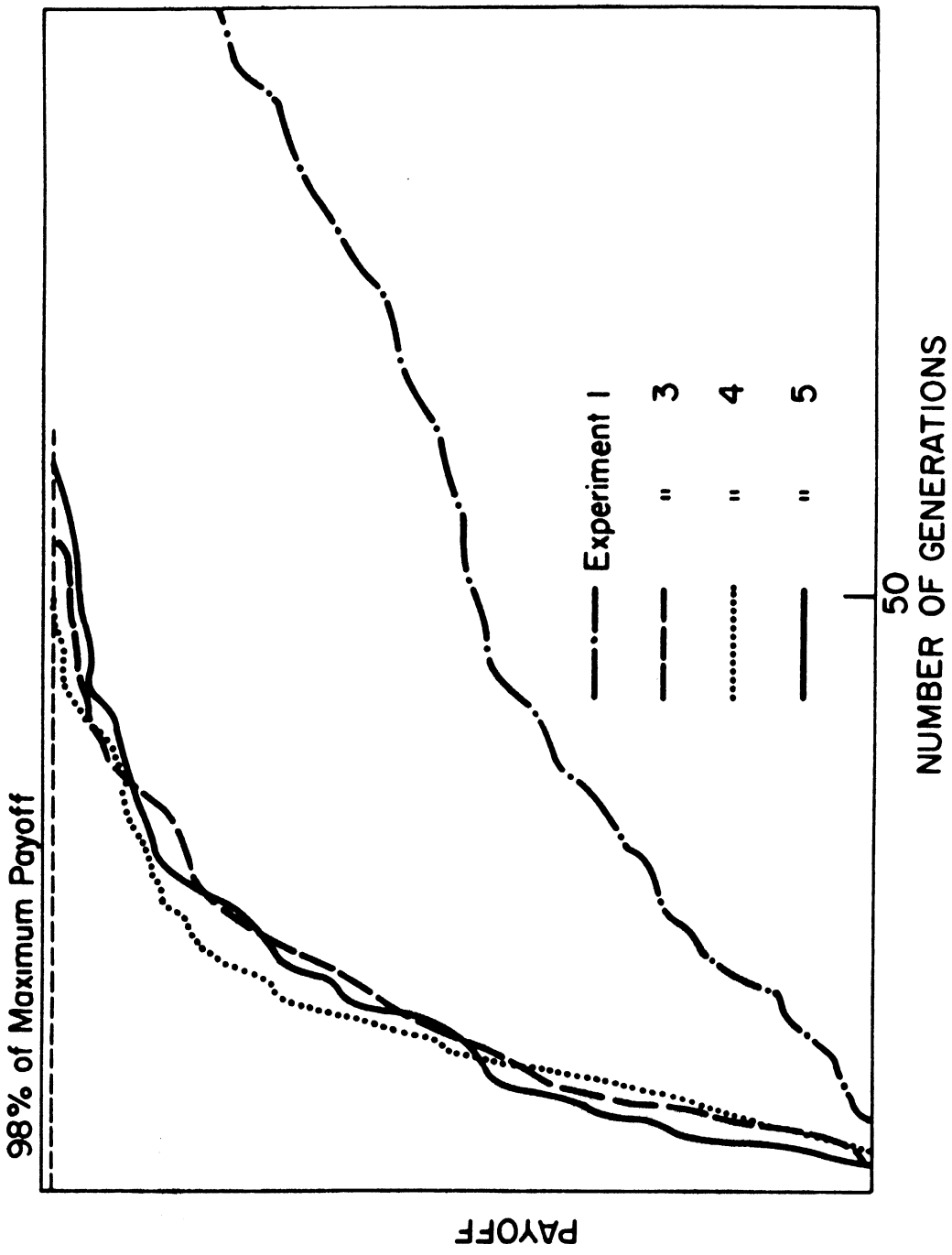


Figure 4.3.1 Sample experiments

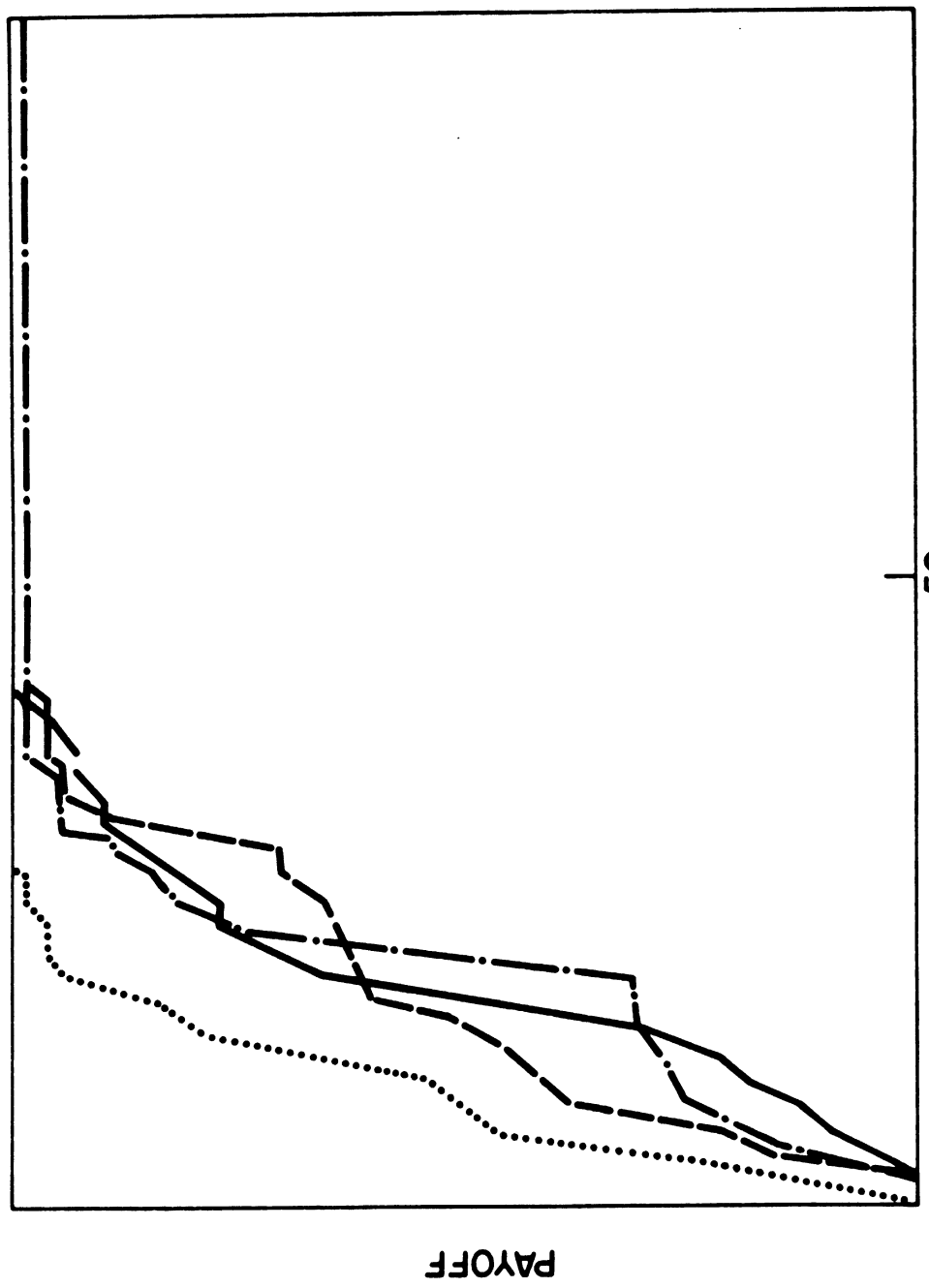


Figure 4.3.2 Sample runs of experiment 4

EX NO	SEL SCM	OPERATOR PROBABILITIES MUT	D.I.	REV	INV	CRS.	% MAX	MEAN	S.D.	SIGN
14	2	.50	.50	.50	.50	.50		65.3	3.76	*
15	2	.20	.50	.50	.50	.50		100.5	26.9	*
16	2		.50	.50	.50	.50	100	150.0	-	

Table 4.3.4 Selection Scheme 2

and processing room. Redundancy (i.e., multiple copies of a subroutine) is important since it guards against loss by mutation. Extra room on the chromosome can serve as a valuable mechanism for accumulating new subroutines. Important subroutines will eventually appear on the left-most region of the chromosome since this results in higher payoff. Thus mutations and crossovers in the right-most region will seldom be harmful. However, if the chromosome is too short, every position would have to contain a crucial subroutine.

At this point we decided to end investigations on this initial task since it did not provide the proper demand characteristics to distinguish between a range of different adaptive plans. However, we have established the following conclusions about reproductive plans:

1) Different selection schemes do effect the efficiency of the plans.
2) Operator probabilities are important in extreme cases. It is necessary to have operators which play different roles and allow the plan to take steps of different sizes in the search space.
3) There seems to be a definite need to have operator probabilities vary during adaptation.
4) Redundancy within the chromosome plays an important role.

These conclusions will be very valuable in future investigations.

The next step in our investigation is to find a task whose optimal performance is very difficult to obtain. Character pattern recognition was chosen for the following reasons:

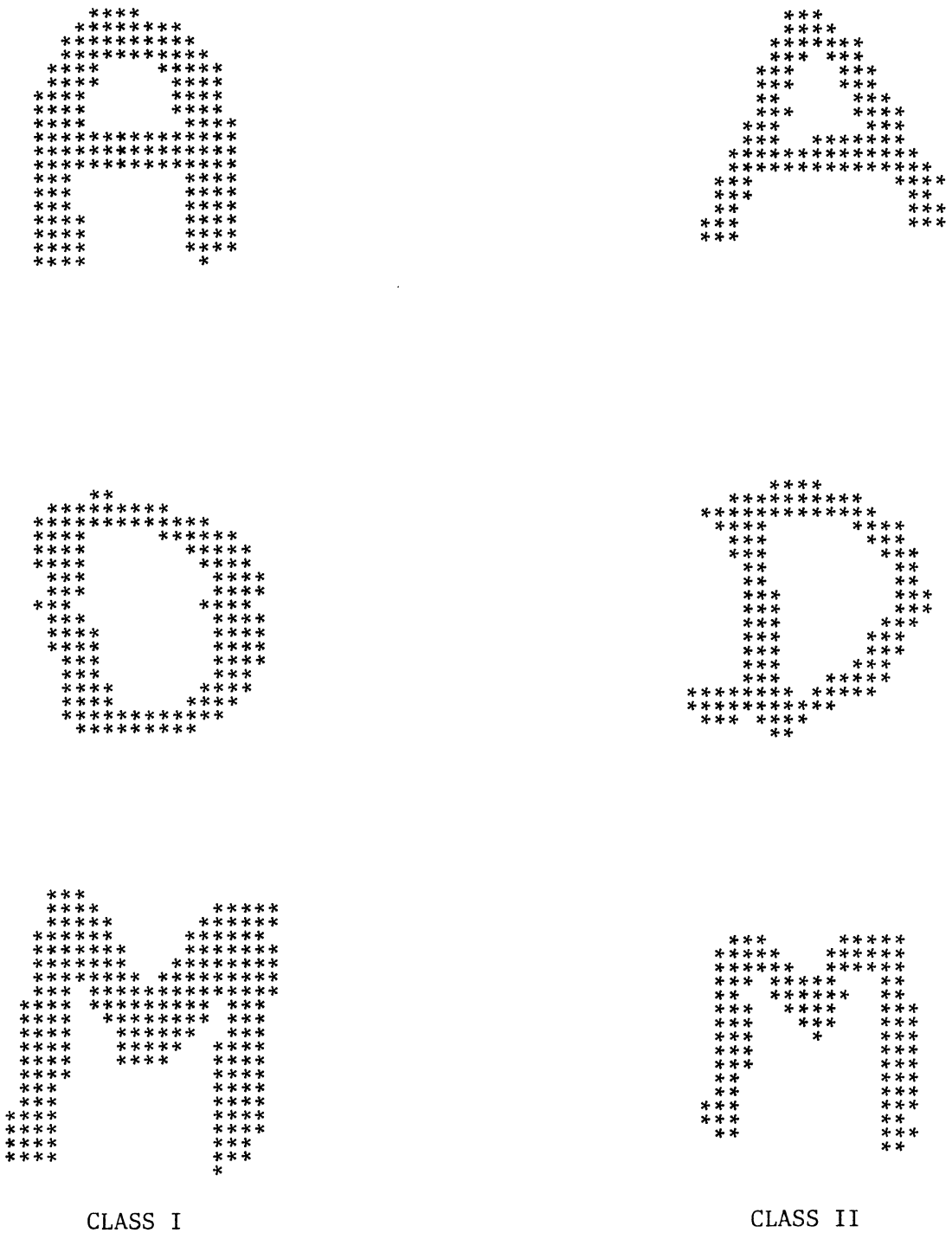
1) The difficulty of the task could be easily but nontrivially varied by using carefully or badly drawn patterns.
2) The field is of interest to artificial intelligence apart from its applicability to adaptive systems.
3) It is a prime candidate for extension by an adaptive system. Despite much research done in the field, many of the problems still remain unsolved.
4) A paradigm already exists which seems applicable with some modifications to the reproductive plans.

Let us now turn to the basic pattern recognition paradigm.

4.4 The Pattern Recognition Task

The pattern recognition task we will use is based on a paper by W. W. Bledsoe and I. Browning [1959]. Patterns are represented on a 25 x 25 grid where each square of the grid contains a 1 if the pattern, when drawn on the grid, intersects that square and a 0 otherwise (see Figure 4.4.1). The information available to the program, however, depends on a specified set of detectors or n-tuples. An *n-tuple* or *detector* is a set of n points or squares on the grid. An n-tuple serves as a primitive detector by returning information which completely specifies the 0-1 configuration of the squares that it investigates. Each different configuration will be called a *state*.

A chromosome for the reproductive plan will consist of a string of



CLASS I

CLASS II

Figure 4.4.1 Sample pattern instances from two different pattern classes. Class distinctions are explained in chapter 5.

detectors. The rest of the pattern recognition scheme will be the same for all devices. As long as the n-tuples are fairly randomly distributed around the grid, the performance of different devices will not vary greatly, due to the redundancies present in most visual patterns. However, this variability is the crucial factor that allows the performance to improve through adaptation.

In order to facilitate discussion we will introduce the following notation. Let all possible n-tuples be ordered and let n_1, n_2,..., n_i, ... designate their names. Each n_i can be regarded as a function which maps an input matrix into an integer designating a state. Let $\bar{n}_i$ be the number of points making up the n-tuple n_i and let n_i^j be the jth state observed by n_i. There are $2^{\bar{n}_i}$ states for each n_i.

The pattern recognition program or device proceeds in the following manner. In its *learning phase*, the program is presented with a grid-pattern along with a name. The program reserves a chunk of memory for this *name*, and then further divides this chunk into slots, one slot for each n_i. In each slot the program records the n_i^j observed for that pattern. If, during this learning phase, another pattern with the same name is input to the program, it is possible that another state n_i^k, $k \neq j$, will also be stored in the n_i slot due to variability within pattern groups. Therefore, it is possible to have more than one state stored in a particular slot under one pattern name.

During the *recognition phase* an unknown pattern is input and a state is observed by each of the n-tuples. Then a comparison is made with each memory chunk for each pattern name. This is done by comparing states recorded in the respective n_i slots of the unknown pattern with those of each known pattern, and totaling all matches. Let M_j be the number of

state matches between the unknown pattern and the known pattern j. M_j can equal at most the number of detectors. The maximum M_j determines which name is chosen for the unknown pattern. This procedure is demonstrated in Figure 4.4.2. Actually, first and second choices are made for each unknown pattern and the number of matches (expressed as a percent) is recorded for each choice. This data will be used later to determine the payoff.

The above scheme was chosen for a variety of reasons. The most important, of course, is the fact that a particular program can be easily changed (by changing some of its n-tuples) without vastly altering its overall performance. Another merit is the program's generality. It is not designed to identify a particular type of pattern but should perform well on a variety of patterns due to the nonspecificity of its detectors. Other reasons for using this scheme include its relative simplicity, speed, and success when compared to other pattern recognition programs reported in the literature. For all the n-tuples used in the tests, $\bar{n}_i$ was restricted to the range $2 \leq \bar{n}_i \leq 6$, due to the fact that larger $\bar{n}_i$'s should result in prohibitively large memory storage.

A device's payoff is determined by how well it recognizes all unknown patterns. For each unknown pattern, a score from 0 to 100 is recorded according to the following procedure. If the correct pattern name is not one of the first two choices, the score is 0. Otherwise *discrimination* is defined to be the difference between the percent of matches (with the unknown pattern) of the first two choices. If the first choice is correct, discrimination is positive; if the second choice is correct, discrimination is negative. The score on each unknown pattern is then a linear function of discrimination, with zero discrimination having a payoff

UNKNOWN PATTERN KNOWN PATTERNS

PATTERN ?		PATTERN 1		PATTERN 2	
N-TUPLE	STATE	STATES	MATCH?	STATES	MATCH?
N_1	3	1,3	YES	1,4	NO
N_2	6	5	NO	3,5,7	NO
N_3	13	10,13	YES	13,16	YES
N_4	1	1,2	YES	1,3	YES
N_5	4	3	NO	5,2	NO
		TOTAL	$M_1=3$	TOTAL	$M_2=2$
		%MATCH	60%	%MATCH	40%

Figure 4.4.2 Sample state matching procedure

A hypothetical case using two known patterns and five n-tuples. In this case the unknown pattern would be taken to be an instance of pattern 1. Note that at least two instances of pattern 1, and at least three instances of pattern 2 have been learned.

of 50 (see Figure 4.4.3). The slope of the score function determines how great a discrimination is required before a perfect score of 100 is obtained. This slope, therefore, constitutes part of the environment. The utility μ is equal to the average score over all unknown input patterns.

It should be noted that the above payoff scheme gives a much more accurate representation of a device's performance than would be the case if one just used the percent correct. For example, a score of about 50 would result from the case in which first and second choices had about the same percent of matches regardless of which choice was correct. This method effectively compensates for many chance indentifications.

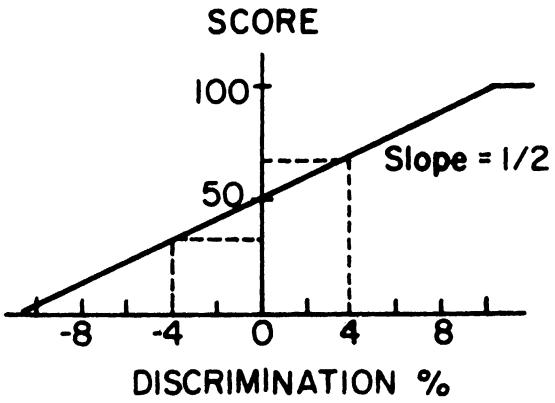
The chromosome or adaptable portion is represented in the follow-in manner:

+5 +372 +9 -518 -213 -35 -76 +44 +348....

Sequential numbers of the same sign indicate one n-tuple while the actual numbers specify the squares in the input grid which the n-tuple observes (input patterns are stored in one dimensional, not two dimensional arrays). A gene is considered to be a whole n-tuple. Therefore, crossover and inversion breakage will occur at n-tuple boundaries where there is a sign change. However, certain types of mutation will make changes within n-tuples. It is not necessary to distinguish between genes and alleles in this task since all genes have the same function.

All chromosomes in the current population will contain the same number of mesh points but not necessarily the same number of genes. The function which translates the chromosome into a working pattern recognition device operates as follows. The chromosome is scanned left to right. After extracting n-tuple n_i, the function sets aside $2^{\bar{n}_i}$ memory locations. Each location contains 16 slots, one for each pattern

Known Patterns	Number of Matches	% of Matches
PAT 1	22	44%
PAT 2	28	56%
PAT 3	36	72%
PAT 4	21	42%
PAT 5	31	62%
PAT 6	34	68%



The program's first choice would be PAT 3 with 72% and second choice would be PAT 6 with 68%. The performance score will depend upon what the unknown pattern actually is. Cases are given below:

CASE I	CASE II	CASE III
Unknown pattern is PAT 3	Unknown is PAT 6	Unknown is PAT 5
Discrimination is +4%	Discrimination is -4%	Score is 0
Score is 70	Score is 30	

Note: If the slope of the discrimination function is changed, the scores in cases I and II will change.. In particular if the slope is $\geq 5/4$, case I will score 100 and case II will score 0.

Figure 4.4.3 Sample payoff determination

name. If available memory (set at 1400 location) is exceeded in the process, then some of the n-tuples on the right-most portion of the chromosome are not used. Chromosomes with an excess of large n-tuples will typically exceed available memory.

The above pattern recognition paradigm is well suited for a first-order plan like a reproductive plan. This stems from the difficulty of arriving at a good measure for ranking individual n-tuples independently of others.

Let us examine a situation which should point out this difficulty. We define an n-tuple to be *completely saturated with respect to some pattern X* if all the possible states of the n-tuple are recorded in its memory slot under the pattern name X. Complete saturation is obviously undesirable since in that case the particular n-tuple does not help at all in distinguishing X from any other pattern. We can make some observations with respect to the saturation phenomenon:

1) The smaller $\bar{n}_i$, the more likely that the n-tuple will become completely saturated since smaller n-tuples have fewer states.
2) The more pattern instances of the same name that are learned, the more likely any n-tuple will become completely saturated.

Obviously an n-tuple must go through various degrees of saturation before it becomes completely saturated.

A critical question is, "What degree of saturation will produce optimal performance?" There does not seem to be an easy answer to this question. Large n-tuples will not become saturated as quickly as smaller ones, but then there is a good chance that the larger n-tuples will not properly match the correct pattern since there are a larger number of possible states. We also might have the case of an n-tuple which in-

vestigates the border regions of the mesh. Such an n-tuple would consistently observe the same state (corresponding to zero or blank mesh entries) for all pattern instances. As such it is the least saturated but probably the most useless detector.

In general, an n-tuple's performance depends greatly on what other n-tuples are present, how many patterns of the same name are learned, and how similar different patterns are. Therefore, it is extremely difficult to evaluate an n-tuple independent of these circumstances. However, in the next chapter we will operate a control experiment which will attempt to extract measures of an n-tuple's goodness. As we have indicated in the previous chapter, the use of such measures involves assumptions and resultant restrictions on a plan's searching ability. In the next few chapters we will show that these assumptions prove to be detrimental for adaptation with our pattern recognition task.

Chapter 5 Nonreproductive Plans

5.1 The Random Selection Plan

This section will serve two purposes. First, we will examine the behavior of the random selection plan on the pattern recognition task and thereby obtain a 0-level performance to which other plans' performance can be compared. At the same time we will test the performance of random plans biased by some unchanging but nonuniform distribution over the set of devices. Secondly, we use the random plan in a number of environments in order to establish what we will call easy and difficult tasks.

The first step is to establish appropriate testing alphabets which will constitute part of the environment. An *alphabet* is a set of pattern instances of the block letters A through P. Two classes of alphabets were created. Class I contains two alphabets of 16 block letters each, hand printed on graph paper and then coded onto cards. Class II contains four alphabets printed on a scope using a light pen routine. In general the two classes differ in the style of the letters, the width of the lines and neatness (Class I patterns were smoother and more carefully drawn). Figure 4.4.1 contains some Class I patterns in one column and Class II patterns in the other. The pattern recognition paradigm contains *no* preprocessing to adjust for differences in pattern size, thickness, translation or rotation.

A particular environment $E\varepsilon\,\mathscr{E}$ consists of a set of alphabets which the device will learn, a different set which it will try to classify, and a mapping between these sets to indicate which pattern instances have the same name. The pattern recognition devices will be tested in various environments.

The random plans will differ with respect to the sampling distri-

bution over the devices. These distributions will be generated by probabilistically restricting the number of n-tuples of certain sizes. This is done for a number of reasons. The most important reason is storage limitations. Large detectors (5 or 6 points) consume an enormous amount of memory due to the exponential growth with n-tuple size. Therefore, we usually desire fewer of these detectors. Secondly, we know from previous work with this pattern recognition paradigm that smaller n-tuples are typically more economical [Bledsoe]. However, when many patterns have been learned, larger n-tuples also become necessary due to the saturation phenomenon discussed above. Finally, we are interested in seeing what effect varying the distribution will have on the performance of the random plan. One usually considers the random plan to sample from a uniform distribution over all devices. However, the critical feature of random plans is that they sample randomly from some constant underlying distribution and do not use any feedback to change their sampling behavior or to modify the underlying distribution.

Table 5.1.1 lists the results of some important experiments which differ in the sampling distribution used. The column labeled PN stands for the probability of an n-tuple with N points. The particular alphabets used for the learning and recognizing phases of the pattern recognition device are designated under the columns "LEARN" and "TEST". The Roman numeral indicates the alphabet class while the Arabic numeral indicates the alphabet within that class.

The first set of experiments (1-7) tested the effect of different sampling distributions. The environment consisted of one learning alphabet and one testing alphabet, both taken from Class I. Mean performance was quite stable over the various distributions. Only

EX NO	N-tuple Distribution P2	P3	P4	P5	P6	Alphabets LEARN	TEST	MEAN	S.D.	SIGN
1	.50	.25	.125	.062	.062	I-1	I-2	47.8	5.07	
2	1.0					I-1	I-2	47.2	5.03	
3		1.0				I-1	I-2	48.3	4.83	*
4			1.0			I-1	I-2	47.8	6.15	
5				1.0		I-1	I-2	46.0	7.28	*
6	.125	.313	.313	.125	.125	I-1	I-2	47.9	6.46	
7	.25	.25	.25	.125	.125	I-1	I-2	46.1	6.36	
8	.50	.25	.125	.062	.062	II-1,II-2,II-3	II-4	58.8	4.93	
9	1.0					II-1,II-2,II-3	II-4	54.3	3.92	**
10		1.0				II-1,II-2,II-3	II-4	58.8	4.68	
11			1.0			II-1,II-2,II-3	II-4	63.0	5.03	**
12				1.0		II-1,II-2,II-3	II-4	60.4	5.13	
13	.125	.313	.313	.125	.125	II-1,II-2,II-3	II-4	61.3	5.39	
14	.25	.25	.25	.125	.125	II-1,II-2,II-3	II-4	60.7	4.89	

Table 5.1.1 Performance of Random Plans Using Different Sampling Distributions

between the extreme cases (Ex 3 and 5) was there a significant difference.

This lack of variation in performance is rather surprising considering the different amounts of memory and different number of detectors generated by the various distributions. Experiments 1 through 4 used about all the detectors on the chromosome, but experiments 2 and 3 used only 45% and 57% of available memory respectively. Experiments 5 through 7 used all of available memory but less than all available detectors. Experiment 5 used only 68% of the chromosome.

We can suggest many reasons why these experiments were insensitive to the different sampling distributions. Many of the n-tuples might be redundant so that their elimination does not adversely effect performance. Although only about half of available memory is used by devices with smaller n-tuples, there are in this case more n-tuples being used. Since only one instance of each pattern was learned in these experiments, the saturation effect is minimal. Therefore small n-tuples are more efficient. Although more memory was set up using large n-tuples, much of this memory was not used.

Different environments, however, produced different results. Experiments 8 through 14 demonstrate performance when three alphabets were learned. In this environment most of the experiments were significantly different from some others. The extreme cases are designated with a double asterisk. The worst performance (Ex 9) is in the case where only 2-tuples are used. It is very likely that saturation played an important role in this case since each n-tuple has only four states and three instances of each pattern are learned. The rest of the experiments suggest that 4-tuples are best suited for this environment.

The next set of experiments (Table 5.1.2) was designed to test the

EX NO	Alphabets LEARN	TEST	MEAN	S.D.	SIGN
15	I-2	II-1	17.5	4.91	*
16	II-1	II-2	43.0	5.28	*
17	II-2	II-3	55.5	5.80	*
18	II-3	II-4	47.5	4.75	*

Table 5.1.2 The Difficulty of Various Environments

difficulty of different environments, given a fixed amount of learning. The n-tuple distribution for all these experiments was identical to that used in experiment 1. All experiments used the same *number* of learning and testing alphabets, but the particular alphabets used were varied. Every one of these experiments produced significantly different results. The worst performance came in the case where the learning and testing alphabets came from different classes, indicating that the class differences cited qualitatively above, are significant. The performance levels in these experiments could be taken as similarity measures between each alphabet pair used.

The purpose of these experiments was to arrive at appropriate environments for testing reproductive plans and to find a suitable n-tuple probability distribution. The n-tuple distribution is important to the reproductive plan since it determines the initial population and affects the behavior of certain genetic operators, such as random mutation.

We decided to use environments involving only one learning alphabet and one testing alphabet for the following reasons:

1) These environments required the least amount of time to

obtain a payoff. Doubling the number of alphabets used doubled the time needed to obtain payoff.

2) The performance levels of the random plan in these environments were significantly lower than the levels in other environments which allowed more learning alphabets. Low performance levels imply a harder task and, therefore, more room for improvement.

3) Performance in this type of environment seemed least sensitive to changes in the n-tuple distributions as evidenced by experiments 1 through 7. This is desirable since the n-tuple distribution will remain fixed when testing reproductive plans. If performance were closely related to this distribution, we would have to run a number of additional experiments to control for this factor. Such experiments would be prohibitively costly.

Having decided on the one-learning-alphabet, one-testing-alphabet type of environment, the choice of an n-tuple probability distribution became less important. One important property was that the distribution provide for some nonzero probability of all size n-tuples. In this case selection can effectively alter the n-tuple size frequency in the population, given that certain size n-tuples consistently result in higher than average payoff. The distribution used in experiment 1 will be used from now on. The environments used in experiments 1 and 15 will be used in most of the remaining experiments. In our discussions the environment in experiment 1 will be called the *easy task* while that in experiment 15 will be called the *difficult task*.

Before ending this section we should say a few words about the

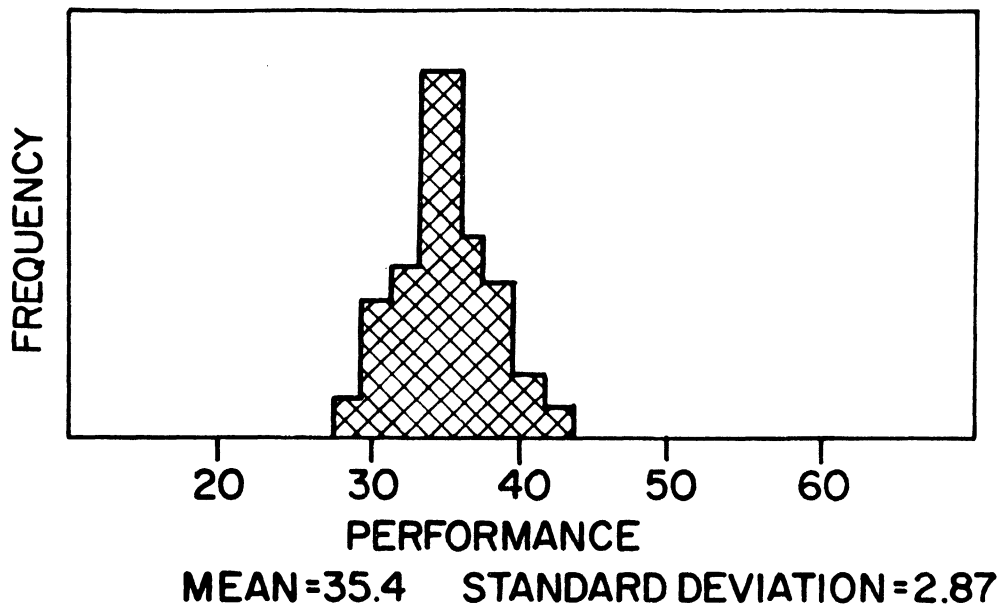
random plans. The larger the variance of the payoff distribution, the longer the random plan will continue to discover significantly higher peaks with successive samples. However, we generally can estimate the true variance with a minimal amount of sampling and thereby estimate the expected number of samples needed to discover an individual within a specified payoff interval.

Figure 5.1 contains a bar graph representation of some distributions generated by the random plan. The two distributions represent the highest and lowest variance produced by different random plans. Most distributions resemble the normal distribution. Therefore, we can estimate that about only one individual out of 1000 will exceed three standard deviations above the mean. For our easy task this performance level is 63.0; for the difficult task, it is 32.2.

Summary

We have now narrowed the range of possible task environments and plans we will consider. In addition we have estimated the performance of the random search plan.

The process of selecting an appropriate task was directed by two criteria. The first was concerned with minimizing real computer running time. This was a practical criterion dictated by experimental or research needs not necessarily related to adaptive systems studies. The second criterion demanded that the task be difficult enough so that significant performance differences between various plans might become evident. By manipulating the number and type of alphabets



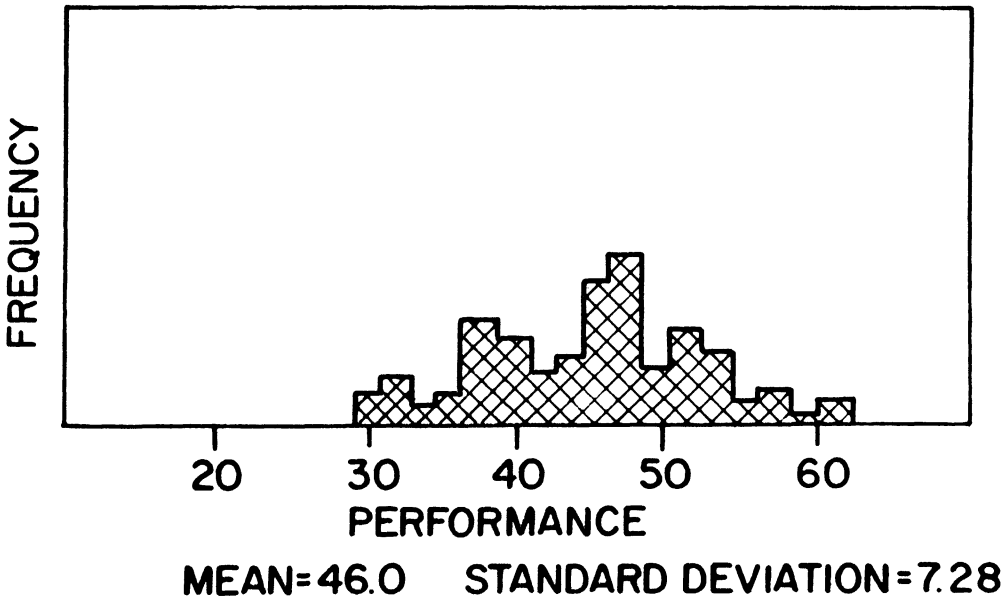


Figure 5.1 Sample payoff distributions for the random plan

used for the pattern recognition task both of these criteria were adequately satisfied.

The possible range of adaptive plans was narrowed by settling on a fixed n-tuple distribution for generating new detectors. This distribution could have varied over different plans; however, initial results indicated that this would not be a worthwhile aspect of plans to study.

Since we obtained the above results by obtaining payoff for a number of randomly generated devices, we have, as a by-product, an estimation of the performance of a random search plan. Nearly all of the performance distributions resembled the normal distribution. The parameters of these distributions enabled us to estimate the probability that random search might select an individual with utility greater than any given level.

The next section will examine very briefly the performance of an adaptive strategy very similar to that used by Klopf. This study will not be exhaustive but is only intended to provide a feeling for what might be accomplished with a nonreproductive plan more sophisticated than the random plan.

5.2 A Detector Evaluation Plan

Before investigating the behavior of reproductive plans we shall look at a nonreproductive plan in order to obtain an estimate of the difficulty of the search space. The only plans available are those which use feedback other than an overall utility in the search process. Unfortunately this choice biases any comparison with reproductive plans, which by their nature are first-order plans. However, we still can obtain some information about the nature of the search space and gain a feeling for the operation of a nonreproductive plan.

The general plan we have chosen is similar to that of Klopf discussed in Chapter 2. Such a plan entails identifying certain inferior detectors in the current device and replacing these with randomly generated detectors. Two methods for generating a random detector were adopted. A partial random mutation, called Mutation I, generates a new detector from a previous one by replacing one randomly chosen point in the n-tuple with a randomly generated mesh point. A complete random mutation, Mutation II, randomly generates a totally new n-tuple by generating n new points. Mutation II is analogous to Klopf's mutation. Mutation I was designed to allow the plan to take smaller steps in the search space. Adaptive plans which modify weights or amplifiers can easily adjust the search step size by varying real valued increments. The problem at hand (i.e., generating new detectors) is more difficult and has been presented less often in the literature.

The problem of deciding which detectors should be replaced or equivalently what characterizes bad detectors is more difficult.

Klopf's criteria involve weights associated with his detector functions or employ a correlation technique. These evaluation methods are not applicable to our pattern recognition paradigm. Our paradigm uses no weights nor do the detectors output numerical values to be used by a correlation plan.

The process of identifying bad detectors for our task relies strongly on intuition. One indication of an inferior n-tuple is the saturation effect discussed in Section 4.4. Having observed many instances of each pattern, a detector becomes saturated and ends up matching all patterns. However, the environments we shall work with involve learning only one instance of each pattern, thereby reducing the saturation effect.

An evaluation procedure often mentioned in the literature is to rate a detector as if it alone were performing the identification task; this obviously entails an independence assumption. In our task a detector records either a match or no match for each pattern. Since only 16 patterns are recognized, scores would be quantized to 16 levels with the majority near the lower end of the scale. With over 100 detectors this would result in many ties for the worst detector. In addition, this method would be displeasing since discrimination, a function of all detectors, and not percent correct is the ultimate indicator of payoff.

Although the above procedure is not particularly satisfying, no real alternative seemed available. Therefore, the above technique was modified a bit to eliminate the difficulty of having only 16 levels in the n-tuple ranking. For each n-tuple the following information

was recorded:

1. The number of times the n-tuple correctly matched the state of the unknown pattern with the stored state for that pattern name in memory. This is the percent correct measure mentioned above.
2. The number of alternate names that were *not* matched, whether or not the correct pattern was matched.

Obviously, a n-tuple which has a low score on both of these counts is very bad since it matches many of the wrong patterns and few of the correct ones. High scores on both these counts identifies a very good detector. However, most evaluations fall between these two extremes and it is not obvious how the above two measures should be combined. The number of correct matches is important only when the n-tuple does not match for incorrect pattern names. In other words, matching all patterns is just as bad as (maybe worse than) matching no patterns. Also, certain nonmatches are more crucial than others due to similarity between patterns. Yet this information is not available until all detectors have been used.

Two evaluation criteria were used in the experiments. Criterion I ranked n-tuples according to the number of correct matches. In case of a tie the second measure, the number of nonmatches, was used to further refine the ranking. Criterion II identified bad detectors using only the number of nonmatches on alternative names. Ties in this case were improbable.

We wanted to come up with a criterion which combines number of matches and nonmatches in a meaningful way. However, no such combination seemed capable of consistently identifying the worst detectors. For example, a weighted sum of matches and nonmatches was a possibility but this criterion would fail to throw out detectors which always matched everything. Such detectors would appear somewhere in the middle of the ranking since they scored high on correct matches. On the other hand, Criterion II would place these detectors at the bottom of the ranking. These worthless detectors were common when n-tuples contained points around the perimeter of the mesh. In this case the n-tuples recorded blanks (zeros) for all patterns and therefore matched all patterns.

The flow chart in Figure 5.2.1 demonstrates the operation of these plans. Initially two randomly generated strings are tested and receive payoff. At this time information is also recorded on the individual detectors. Depending upon which mode the plan is using, either the best or the newest string is saved as the current string. Then a specified number of the worst detectors (ranked by a specified criterion) are modified using a specified mutation operator. This process is then iterated. The type and number of mutations and the evaluation criterion remain fixed over an entire experiment. All experiments operated on the difficult task.

Table 5.2.1 presents results using Criterion I. The column headings are as follows:

EX NO ----- Experiment identification number

RN NO ----- Run identification number

MUT ----- Mutation type and number of detectors mutated

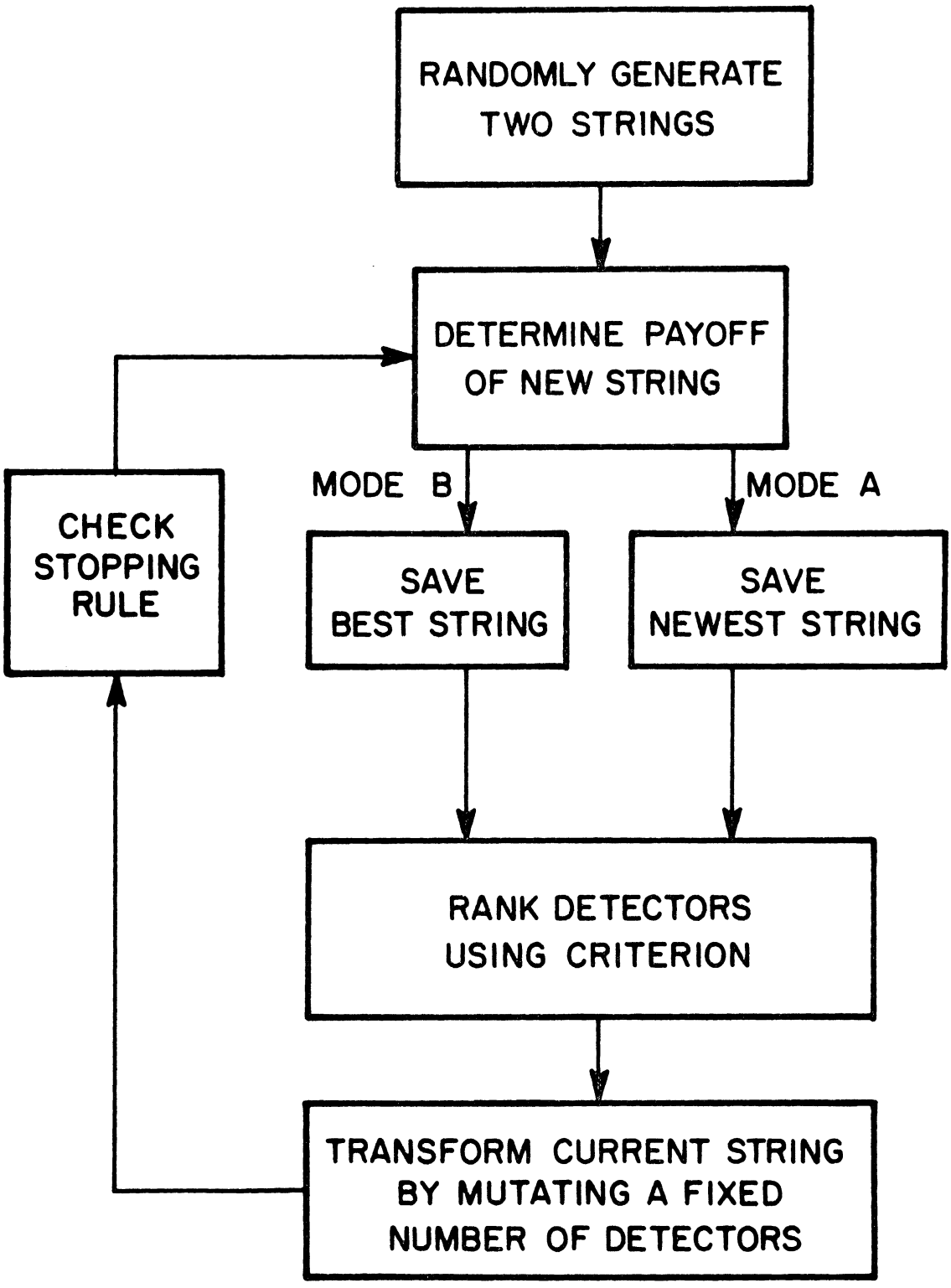


Figure 5.2.1 Summary of the detector evaluation plan

EX NO	RN NO	MUT	MOD	PAYOFF AT CERTAIN TIMES 100	200	400	600
1	1	II-1	A	26.4	27.8	37.5	35.5
	2			25.5	30.4	28.4	30.7
2	1	II-1	B	26.7	27.5	37.7	52.4
	2			37.2	40.2	45.6	45.6
3	1	I-1	B	22.2	22.2	22.2	22.2
	2			23.8	23.8	23.8	23.8
4	1	II-3	B	40.8	40.8	40.8	43.1
	2			38.6	43.4	43.4	43.4
5	1	II-5	B	36.4	37.2	37.2	45.1
	2			50.4	51.3	51.3	51.3
6	1	I-3	B	31.1	37.1	45.4	52.4
	2			24.9	25.2	49.9	50.6
	3			22.7	23.8	24.7	28.0
	4			21.6	21.6	21.6	21.6
		Average		30.6	32.3	36.3	39.0

Table 5.2.1 Results: Detector Evaluation Criterion I

MOD ---- Mode of operation. A, save newest string.

B, save best string.

Since the object of these experiments was to get a feel for the detector evaluation plans rather than to converge on the best detector evaluation plan, most experiments were run only two times.

Only the first experiment operated in mode A (saving each new string). This mode eliminates the problem of false peaks and is similar to that used by Klopf, and Uhr and Vossler. However, the results with this scheme were not favorable. Maxima discovered at one time were lost and not attained again later. The average maximum of each run was 38.9 while the average payoff of the current string

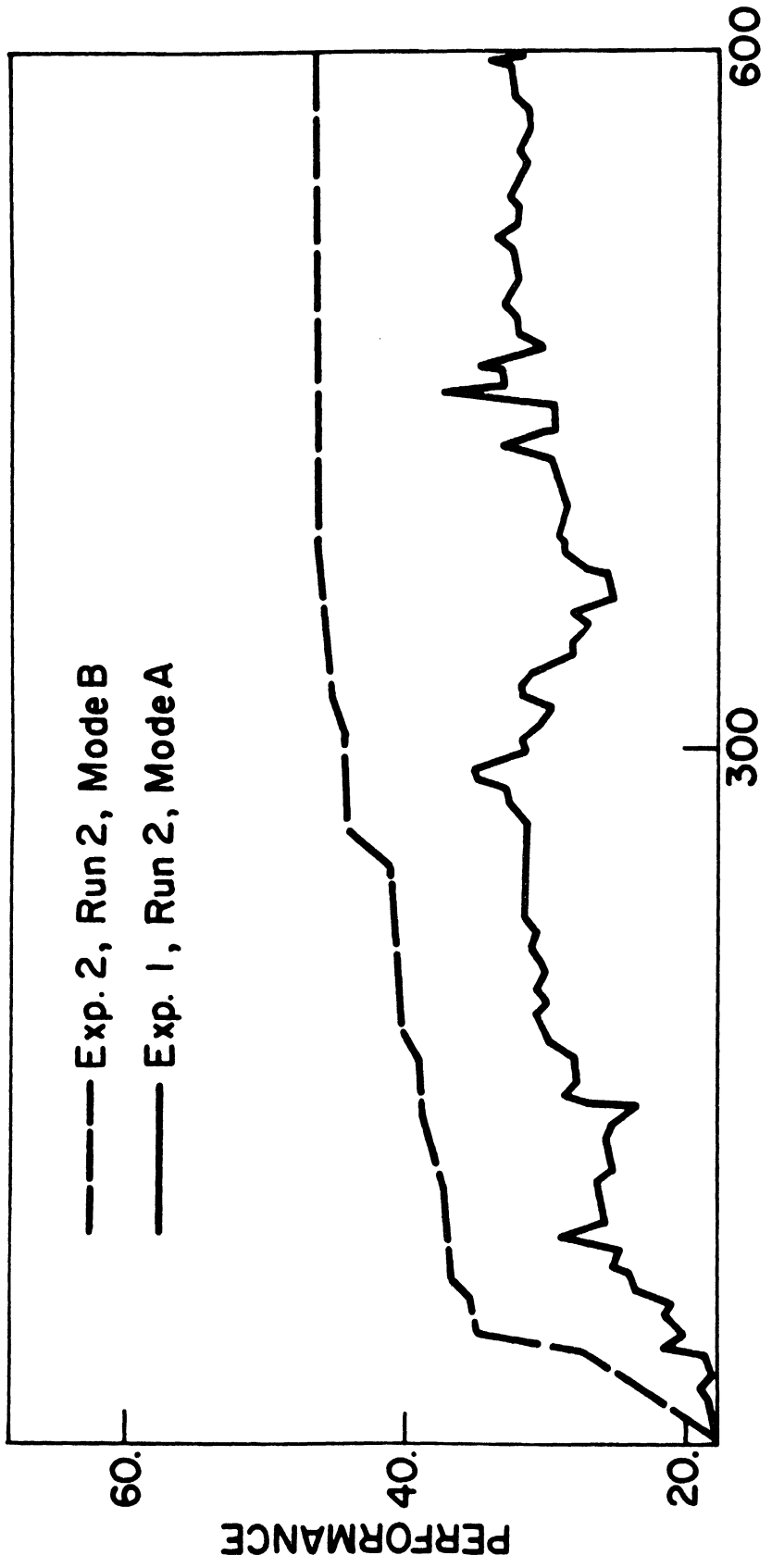


Figure 5.2.2 The effects of saving the newest and best strings

after 600 samples was only 33.1. This type of performance might indicate that Criterion I was not correctly designating the worst detector with respect to the payoff. Yet even if it did designate the worst detector in the string, it is still possible that mutation could generate an even worse detector especially when performance was fairly high. Figure 5.2.2 shows a plot of comparable runs using mode A and mode B.

The remaining experiments with Criterion I operated in mode B since mode A was obviously inferior. The experiments tested the effect of the type of mutation (I or II) and the number of mutations per string (1, 3 or 5). The main characteristic of all the experiments was the tendency of the plan to land on a false peak which it subsequently could not get off of. Many of the runs reach these peaks in less than 100 samples. This behavior effectively wasted most of the samples. No experiment was superior to any other based on the few runs which we made. Experiment 6 appeared to be consistently producing superior performance, but additional runs demonstrated that it was just as susceptible to false peaks as other experiments.

It was hoped that many of the failings of the first set of experiments would be corrected by using Criterion II to evaluate the detectors. However, the results, displayed in Table 5.2.2, were no better. The average performance of all runs using Criterion I was not significantly different from the average performance using Criterion II.

The best performance observed so far has been 52.4. This is about seven standard deviations above the random sampling mean of 17.5 on

EX NO	MUT	MOD	PAYOFF AT CERTAIN TIMES 100	200	400	600
7	II-1	A	18.7	18.7	27.0	21.9
8	II-1	B	34.5	38.5	43.5	46.3
9	I-1	B	29.2	37.9	45.3	49.7
10	II-3	B	30.1	35.4	35.9	35.9
11	II-5	B	18.7	18.7	18.7	18.7
12	I-3	B	21.0	23.0	29.7	29.7
				Average		33.7

Table 5.2.2 Results: Detector Evaluation Criterion II

the difficult task. Considering that the plan used only 600 samples, it is far superior to the random plan. However, some runs ended up with performance levels in the low 20's after 600 samples. This performance is inferior to that of the random plan. One would expect a good adaptive plan to consistently operate better than the random plan.

We now have a better indication of the difficulty of the search space. Most runs reached a performance level of about 35.0. We also have an indication that some detectors play a major role in the operation of a device. Often a device would realize an immediate increase in performance due to the mutation of a single n-tuple out of a total of 110.

Summary

The purpose of this section has been to test the performance of nonreproductive plans which commonly appear in the literature. We have not attempted a complete evaluation of these plans but have restricted ourselves to some obvious implementations.

These plans proceeded in a serial manner; sampling was dictated by the evaluation criterion. Therefore, most of the new samples were concentrated in a particular region of the space and adaptation proceeded in a fairly rigid manner. Were the detector evaluation criterion very good, this search method could have proven very efficient provided that there were no strong interactions between detectors. However, it was difficult to come up with a good detector evaluation criterion. As a result, some of the runs were inferior to the random plan and most spend substantial amounts of time on false peaks.

One might argue that a much more sophisticated detector evaluation technique would have resulted in a marked improvement in the plan's operation. Such techniques might involve checking a detector's performance with respect to certain crucial patterns or checking for redundant actions by detectors. However, the plan still could prove inferior to other plans. As mentioned above, strong nonlinear dependencies between detectors could complicate the action of any scheme which evaluates and modifies single components. But the cost factor may be even more important. By extracting the extra feedback information required by Criterion I or II, the plan consumed twice the amount of time to run the pattern recognition program. Since this process constitutes most of the adaptive plan's operation time, we must be

willing to allow a first-order plan twice as many samples in order to approach a fair comparison. A more sophisticated evaluation technique would require even more time, not including the fact that a whole series of experiments would have to be run for each new task before a good evaluation criterion were discovered. Therefore, we can strongly support putting more work into the adaptive plan rather than trying to extract detailed information on each new device about which we often know very little.

Reproductive plans proceed in a parallel manner; sampling is directed in a general way by specifying selection schemes, recombination operators, and operator probabilities. In the rest of our work we will demonstrate that reproductive plans can operate efficiently in addition to discovering substantial peaks in the search space.

Chapter 6 Reproductive Plans: Preliminary Investigations

6.1 Investigation Techniques

In this chapter we shall begin our search for a good reproductive plan. An adaptive plan is responsible for conducting a search in the set of devices $\mathscr{A}$; the process of finding a good plan reduces to a search in the set of plans $\mathscr{T}$. However, the set $\mathscr{T}$ differs from $\mathscr{A}$ in that $\mathscr{T}$ is usually not completely available beforehand as a set of well-defined objects. Rather $\mathscr{T}$ is generated as the search takes place and as needs dictate. Certainly small subsets of $\mathscr{T}$ are identifiable beforehand as might be the case if part of a plan were determined by a certain parameter setting. However, most interesting new plans are created through the use of new techniques rather than discovered by converging on an optimal set of parameter values. This is one reason why we might not be able to use another adaptive plan to search the space $\mathscr{T}$, although this certainly is a possibility in many cases. In fact, this type of a procedure was suggested by Newell *et al.* in their work with GPS.

In our case as in most others, new adaptive plans will evolve through intuitive and heuristic means. In investigating the behavior of certain plans, we will try to pinpoint failings and inflexibilities. New plans will hopefully correct these observed deficiencies and eventually perform better as measured by our quantitative criterion χ.

There is one difficulty with our search technique which we will have to live with. This involves the recurring problem of interdependencies. The plans we shall be working with have well-defined components such as the population size, the selection scheme, and the genetic operator probabilities. The substitution instances of these components generate

a subset of $\mathcal{T}$ which although small when compared to typical search spaces, cannot be searched in any great detail due to the tremendous cost involved. It typically takes hours of computer time just to evaluate one reproductive plan operating on our task. Therefore, we will usually resort to the practice of using intuition whenever possible to reduce this subset of $\mathcal{T}$ even further. At this point we will try to choose a region where we feel there is a minimum of interaction between components and then proceed to modify components independently. When in doubt we shall repeat the tests in other regions of $\mathcal{T}$.

One certainly might criticize this procedure in light of what we have said previously about independence and linearity assumptions. However, when costs are very high as is the case in our search of the space $\mathcal{T}$, there is really no alternative unless we restrict ourselves to a very small subset of plans. We feel that our method will be more fruitful. We might mention that there is very little empirical work in the literature which addresses itself to this problem of searching a space of adaptive plans.

There are other difficulties which we must consider in our first attempts at searching $\mathcal{T}$. One of these involves the use of a stopping rule for the maximum utility criterion. Letting plans run for a fixed number of generations has the virtue that all comparisons are based on an equal number of samples. However, this procedure has its faults in a high cost situation. Primarily we are reluctant to continue sampling when little or no gains are being made. This situation may arise after a plan has become stuck on a false peak or perhaps when a very good plan has found a significant peak very quickly. Also a fixed generation

stopping rule often fails to distinguish superior plans. Plans are usually good due to their ability to maintain a steady performance increase over many generations. If all plans are stopped too early they may all appear to have maintained good performance and may have obtained similar performance levels. On the other hand, if all plans are allowed to run for an extremely long time, the inefficient plans may eventually manage to reach a significant peak while the efficient plan has been on that peak for many generations.

The marginal utility (slope) stopping rule would alleviate a few of these difficulties. In this case all runs would proceed until the marginal gain in performance over the last few generations falls below a certain minimal level. However, this rule is not without faults either. First of all comparisons between plans would be made using an unequal number of samples. Secondly, one still must decide on an acceptable marginal utility level. Finally, this rule may terminate a run which is only temporarily stuck on a false peak but which still has the potential for further performance increases.

A third stopping rule is also possible. The *payoff stopping rule* would require all runs to proceed until a specified payoff level had been reached. Comparisons are then made on the number of samples required to attain this level. This stopping rule was used in Section 4.3 on the initial task. Its main fault is that its use requires *a priori* knowledge of what constitutes a significant yet reasonably attainable performance level. Furthermore, it stresses only efficiency, assuming that all plans can reach a certain level given enough time.

The initial process of searching $\mathcal{T}$ will entail trying out a number of stopping rules in our comparison of different plans. Hopefully, we will eventually settle upon one rule which can be effectively used in further investigations. However, we may benefit by evaluating plans with respect to a number of criteria, especially when two plans seem indistinguishable when examined with respect to only one criterion.

One additional problem we will have to resolve fairly early is the choice of a task environment. The easy task would probably require less running time since there is less room for improvement. On the other hand, this task might not be able to discriminate between two very good but significantly different plans. We encountered this problem with the initial task which only served to point out the very bad plans.

Hopefully, many of these problems will be resolved fairly early in our investigations so that we may spend most of our time investigating different aspects of reproductive plans. The rest of this chapter will attempt to resolve some of these technical problems while investigating two important components of reproductive plans: selection schemes or the method of allocating new offspring and the method of using genetic operators.

6.2 Selection Schemes

In this section we will examine some initial runs of our reproductive plan and begin our search for some good plans. This search will entail developing a good method for determining how offspring should be distributed among the population members. As we discovered in Section 4.3, selection schemes can have a significant effect on a plan's performance. These schemes directly affect the search process since they bias the regions from which new samples are obtained by directing the duplication of population members. A scheme which overemphasizes very good individuals may induce a narrow search in a small region of the space. The net result is usually a decrease in the population variance, a measure of the variety of genes and individuals in the population. Without this variation the evolutionary process becomes very inefficient.

However, a scheme which duplicates most individuals equally, giving little regard to their performance, may also prove inefficient. Our investigations on the artificial task in Section 4.3 showed that a scheme of this type was inferior to one which used performance measures to influence the duplication process.

We will now investigate the performance of four selection schemes operating on the difficult and easy task using two different population sizes. Let us adopt the following notation for discussing population sizes. An M/N population consists of M individuals, of which the N best remain as *population members* after each generation to possibly produce offspring. M - N offspring are produced each generation. Initial experiments will use a 20/10 or a 12/6 population.

The recombination portion of the reproductive plan will not vary

through the initial experiments on selection schemes. The genetic operators we will use are crossover (single breakage), inversion and the two mutation operators used in the detector evaluation plan in Section 5.2. During recombination, the duplicates of individuals are randomly paired. Crossover is applied to each pair with probability .5 while inversion is applied to each duplicate with probability .5. Then each new individual receives from 1 to 5 mutations of each type, the specific number being drawn from a uniform probability distribution.

Before the regular reproductive plan was tested, we tried a modified version which does not guarantee a monotone increase in the member population's performance. After each new set of offspring are produced the parent or member portion of the population is discarded leaving only the offspring to carry on the evolutionary process. This version failed to produce any significant adaptation using the 12/6 population. With a much larger population there is a good chance that many of the offspring will closely resemble many of the more superior parents and thus maintain a similar level of performance. However, with relatively small populations and an even smaller number of offspring per generation, this version is ineffective. As we later found out, less than one-third of the offspring typically perform well enough to assume a position as a population member. Therefore, the best of the past performers must be maintained until they can be replaced by even better individuals.

Two selection schemes, developed along the lines of those used on the initial artificial task, were used for the first few experiments:

Scheme 1: The basic scheme -- the payoffs of the population members are

decreased by the minimum payoff of this group and these adjusted values are used as weights to determine duplication rates.

Scheme 2 : Payoffs of the population members are used directly as weights to determine duplication rates.

Selection schemes are basically allocation of resource techniques, the "resources" being the number of offspring permitted at each generation. Each individual is duplicated according to how its weight compares to the sum of all weights. If all weights are very close in value, it may happen that each individual is alloted only a fraction of a duplication. If all offspring slots have not been allocated after the duplication rates have been rounded to integers, then randomly chosen individuals which have not already been duplicated are reproduced.

The first few experiments were run for 50 generations. Four or five statistical runs were made for each experiment using, however, the same initial population. A summary of results using two different selection schemes, population sizes, and tasks, appear in Table 6.2.1. The notation under TASK stands for easy (E) or difficult (D) task. The values under MEAN are the mean performance values of the *best* individual in each run after 50 generations.

Surprisingly, there was no significant difference between experiments differing only in the selection scheme used. As one may recall, the basic scheme proved to be superior on the artificial task. However, those results were based on a 25/8 population, in which case there are more offspring slots than population members.

EX NO	SEL SCM	POP SIZE	TASK	MEAN	S.D.	SIGN
1	1	12/6	E	75.1	2.5	
2	2	12/6	E	77.3	2.3	
3	1	20/10	E	78.8	2.7	
4	2	20/10	E	79.2	2.4	
5	1	12/6	D	50.1	3.7	
6	2	12/6	D	51.7	2.0	
7	1	20/10	D	53.6	2.0	
8	2	20/10	D	55.2	3.5	

Table 6.2.1 Initial Experiments on Selection Schemes

Upon closer examination of the actual duplication rates, we found that Scheme 2 resulted in duplicating each individual exactly once. This meant that no selection took place with this scheme after the current population had been determined. On the other hand, the basic scheme seemed to tend toward overselection; with this scheme the best population member often contributed to 50% of the offspring. This effect obviously became more serious when one individual remained the best population member over many generations.

It seemed a bit puzzling that these two schemes did not turn in significantly different performance. However, it is possible that they both reached similar performance levels through different types of inefficient operation. Presumably a scheme which combined aspects of both would produce better performance levels. The following schemes were designed to improve upon the basic scheme by checking for over-selection situations:

Scheme 3: The basic scheme is used with the constraint that a maximum of two duplicates per individual per generation is permitted. If this constraint is exercised, then randomly chosen popu-

lation members are duplicated to fill the remaining offspring slots, as long as no one individual is duplicated more than twice.

Scheme 4: Scheme 3 is used with the additional constraint that a maximum of three offspring per individual is permitted over all generations. However, this constraint is relaxed when excess offspring slots are being randomly distributed.

Most of the experiments were rerun using the new schemes. In addition, the difficult task was run up to 100 generations and then continued under a marginal utility stopping rule up to a maximum of 125 generations. The slope stopping rule allowed runs to continue as long as there was a net performance gain of 1.0 in the previous 15 generations.

Table 6.2.2 summarizes these new experiments in their relation to the past experiments which have been repeated for ease of reference. GEN 50 and GEN 100 indicate mean performance after 50 and 100 generations. STOP MEAN and STOP GEN represent the mean performance and mean generation when the slope stopping rule was exercised.

With a number of experiment pairs both Schemes 3 and 4 were shown to be significantly superior to the others. In the first group of experiments operating with a 12/6 population on the easy task, Scheme 3 outperformed both Scheme 1 and 2 while Scheme 4 outperformed Scheme 1. The next two groups of experiments showed no significant difference along the lines of the first group. However, on the basis of the slope stopping criterion, Scheme 2 dominated Scheme 1 at the 10% level. This

EX NO	SEL SCM	POP SIZE	TASK	GEN 50	GEN 100	STOP MEAN	STOP GEN	SIGN
1	1	12/6	E	75.1	--	--	--	
2	2	12/6	E	77.3	--	--	--	
9	3	12/6	E	80.4	--	--	--	*
10	4	12/6	E	79.1	--	--	--	*
3	1	20/10	E	78.8	--	--	--	
4	2	20/10	E	79.2	--	--	--	
11	3	20/10	E	78.9	--	--	--	
12	4	20/10	E	80.2	--	--	--	
5	1	12/6	D	50.1	56.0	50.5	74	*
6	2	12/6	D	51.7	58.6	56.6	79	*
13	4	12/6	D	49.4	54.7	54.7	87	
7	1	20/10	D	53.6	62.7	59.7	87	
8	2	20/10	D	55.2	63.9	62.5	102	
14	4	20/10	D	59.3	64.7	61.8	79	*

Table 6.2.2 Summary of All Selection Scheme Experiments

reinforced previous suspicions that Scheme 2 was better than Scheme 1 since the performance of Scheme 2 always dominated (but not significantly) that of Scheme 1.

The fourth set of experiments repeated the trends of the first set. Based on performance after 50 generations, Scheme 4 was definitely superior to Scheme 1 and dominated Scheme 2 at the 10% level. However, these differences disappeared by the 100th generation, indicating that the other plans were eventually able to catch up although they were not as efficient. This lack of efficiency is demonstrated again in considering performance using the slope stopping criterion. Although all three schemes reach performance levels which were not significantly different, Scheme 2 required significantly more samples to do this.

Based on these experiments we can generally conclude that Schemes 3 and 4 are both superior to Schemes 1 and 2. Figure 6.2. summarizes all significant result in graphical form. The circles designate selec-

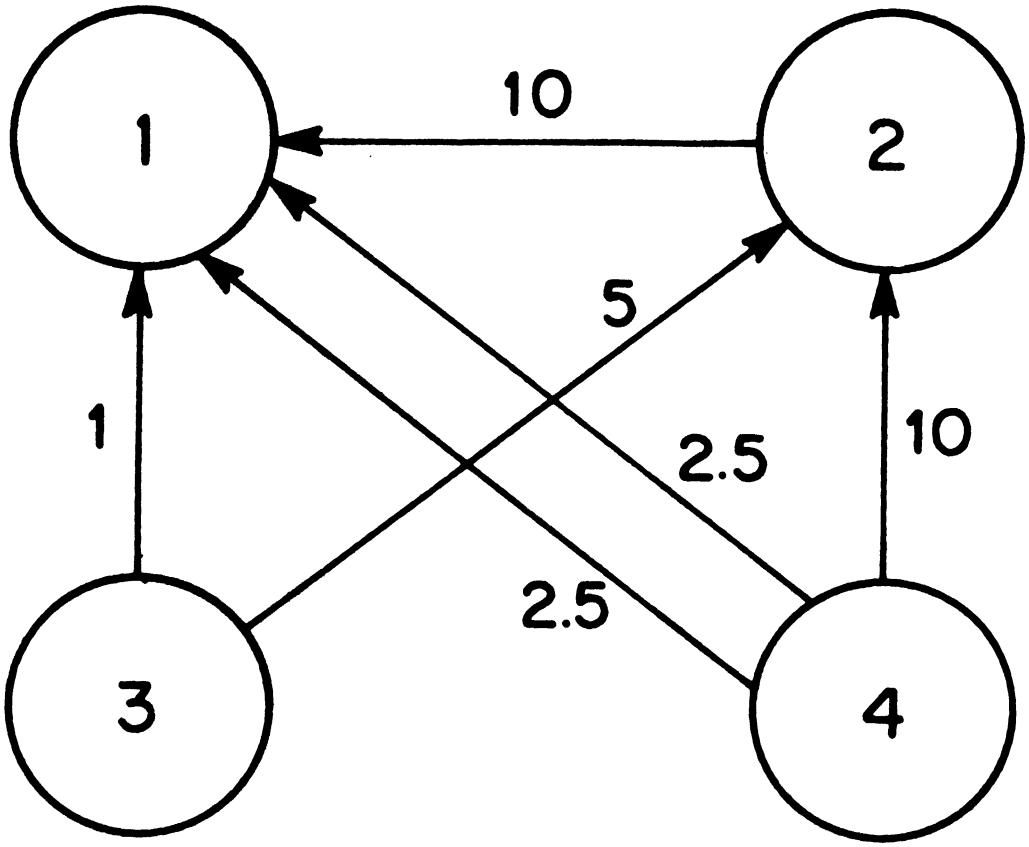


Figure 6.2 Summary of significant results between selection schemes

tion schemes while each arrow designates a test which distinguishes between two schemes. Labels on the arrows indicate percent significance levels.

Scheme 4 has been chosen for use in further experiments. There have been no results in this section to indicate which of Schemes 3 or 4 is superior. However, Scheme 4 seems more intuitively appealing since it places an upper bound on the total number of offspring an individual may produce. Furthermore, by allowing this upper bound to vary, one can consider Scheme 3 a special case of Scheme 4.

As we mentioned earlier in our discussion of search techniques, some tests will be reapplied in different regions of the space to determine if the results were overly dependent on other components of the plan. Since the results on selection schemes have not been as clear cut as one might desire, we have repeated some of these tests after we have developed what we consider to be one of the best reproductive plans. The details of this plan will be discussed later as it is developed. However, it seems appropriate here to present the results of this plan varying only the selection scheme. This will firmly establish the superiority of Scheme 4.

Table 6.2.3 summarizes these results. After 85 generations Scheme 4 dominates Scheme 2 at the 10% level. However, using the slope stopping criterion, Scheme 4 dominates Scheme 2 at the 0.5% level; there were no significant differences in the number of generations over all experiments. There were no significant difference in means between experiments 15, 17 and 18 using the slope stopping criterion since the variances of experiments 15 and 17 were so large. In fact the variance using Scheme 4 differed from that of Scheme 1 at the 1% level and that of Scheme 3 at

EX NO	SEL SCM	GEN 85	STOP MEAN	STOP GEN	STOP S.D.	SIGN
15	1	67.1	73.1	134	7.2	
16	2	66.6	69.5	114	3.2	
17	3	68.2	71.3	116	5.1	
18	4	70.4	76.1	140	1.3	*

Table 6.2.3 The Effect of Selection Schemes on a Very Good Reproductive Plan

the 2% level. This test along provides sufficient reason to choose Scheme 4 since a small variance *between runs* is a desirable quality of adaptive plans. In practice one will make only one run of a plan and assume this run to be fairly representative.

Summary

We have now begun our search for a good reproductive plan by trying our various selection schemes. The selection scheme is an important aspect of the reproductive paradigm since overselection may result in false peaks while underselection may result in slow, inefficient adaptation. The scheme finally settled upon was basically a payoff-weight scheme modified so that overselection would not result. It should be noted that the selection scheme effectively narrows the set of devices upon which the function $\tilde{\tau}$ will operate. This is an important step in directing the search of the space $\mathscr{A}$.

6.3 Analysis of Initial Investigations

Besides serving to identify a good selection scheme, the experiments in this chapter also represent our first attempts at using a reproductive plan on our pattern recognition task. The average performance of the reproductive plan operating on the difficult task after 600 samples (12/6 population after 100 generations) was 56.4. This compares very favorably with the average of 39.0 for the detector evaluation plan. The range of performance values over all runs was 12.3 for the reproductive plan compared to 30.8 for the detector evaluation plan. Furthermore all runs with the reproductive plan produced devices with performance levels at least 6 standard deviations above the random sampling mean indicating performance far superior to the random plan. Yet 28% of the runs with the detector evaluation plan produced maximum performance within two standard deviations of the random sampling mean. These comparisons indicate that our reproductive plans are definitely producing significant adaption using only first-order feedback.

These initial experiments have also demonstrated the importance of choosing a good criterion χ. It was felt that the maximum utility criterion was the best, yet it was not obvious what was the best stopping rule. Comparisons based on performance after a fixed number of generations proved very useful especially when such comparisons were made before performance of some plans began to level off. However, comparisons using the slope (marginal utility) stopping rule also proved helpful in testing a plan's ability to maintain efficient

adaptation. Even when no performance difference resulted, we sometimes found a difference in the number of samples used. This use of multiple criteria seems to be a good way to resolve the problems involved with using a single criterion since we now can simultaneously select for plans which are both efficient and effective.

The experiments in this chapter will also help to direct future investigations. It seems likely that the easy task will not prove too useful in the future in distinguishing the effectiveness of different plans. Mean performance levels are approaching 80 while there is evidence that maximum performance for this task is around 87. This does not leave sufficient room to significantly distinguish many different plans. The difficult task costs more to operate since more generations are usually needed before performance levels off. However, this often provides good plans with a better opportunity to distinguish themselves.

Comparing experiments which differ only with respect to population size, we find no significant difference. Table 6.2.2 seems to indicate that the 20/10 population is better than the 12/6 population when operating on the difficult task. However, this comparison is based on an equal number of generations. When comparisons are based upon an equal number of samples we find there is no difference in performance. Furthermore, comparisons on the easy task after 50 generations also produce no significant difference even though the 20/10 population has received 66% more samples than the 12/6 populations. Most of the future runs will use populations which sample only six new individuals per generation. However, the benefits of a large member population

Formal Element	The basic reproductive paradigm	The addition of Selection Scheme 4
M_t	Payoffs of current population	Add: Total number of offspring for each population member over generations
$\tilde{\tau}$	Reproduction and recombination	Change reproduction (sampling scheme) to avoid over or under selection
m	Save payoff of new individuals	Update total number of offspring

Table 6.3 Changes Due to Selection Scheme 4

can still be realized by running a 20/14 or a 40/34 population. Typically the poorer members of these large populations will not be selected to produce offspring. However, if a set of individuals tend to retain possession of the top positions in these populations, they soon will have exhausted their maximum allowable number of offspring and the remaining offspring will be randomly distributed among the other population members. This is a technique of selection scheme 4 which is not available with schemes 1, 2, or 3.

Table 6.3 indicates how the formal elements of an adaptive plan, M_t, $\tilde{\tau}$, and m have been modified or extended as the result of incorporating selection Scheme 4. Figure 6.3 points out the aspects of our reproductive plan that have been changed, investigated or further specified

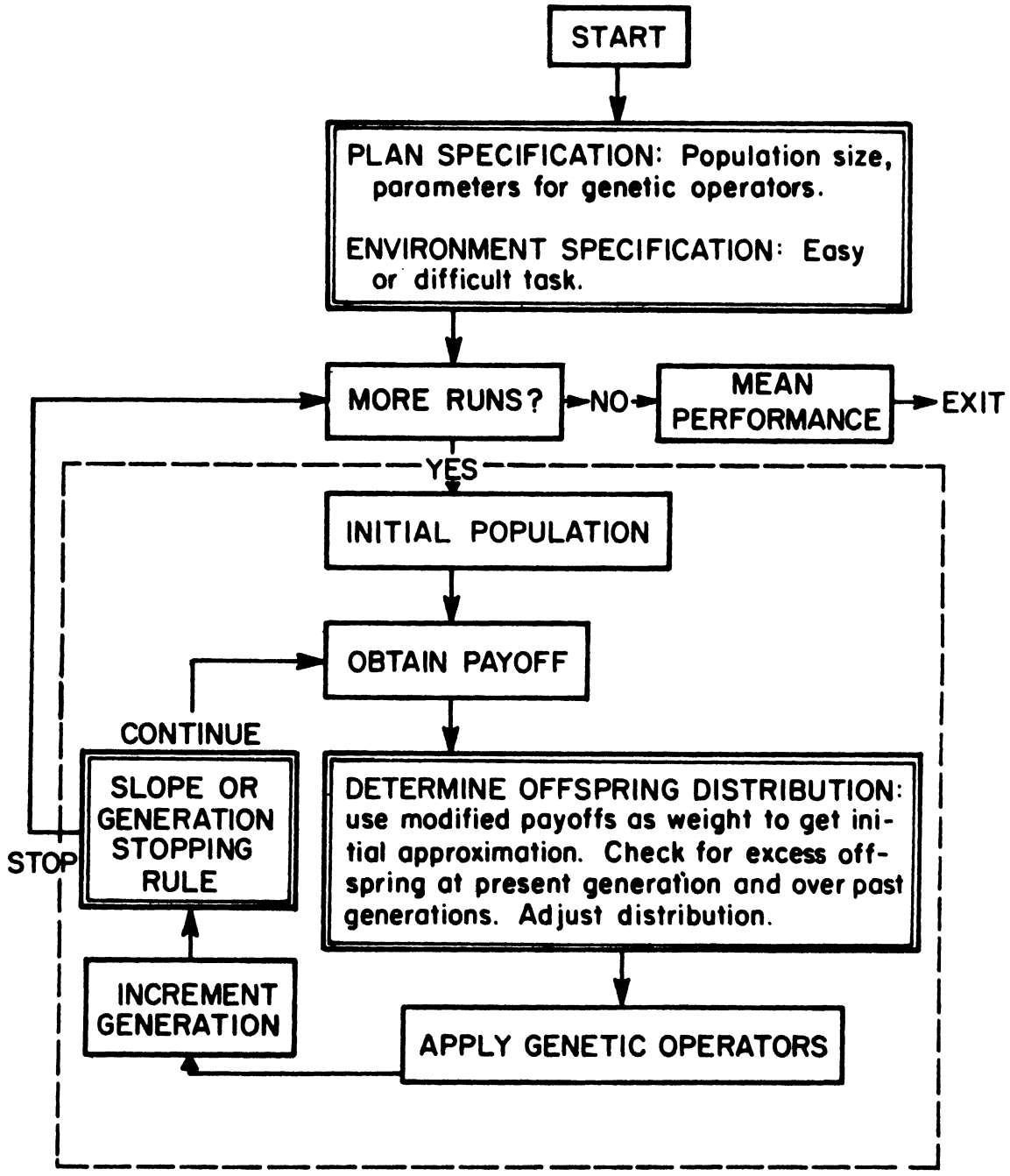


Figure 6.3 Current status of the investigation

as a result of the experiments performed in this chapter. It will be the practice in these figures to abbreviate operations which have not changed since the last figure and to put a double box around operations which have changed or have been specified in more detail.

The next chapter will continue our search for a better reproductive plan by investigating the roles of various genetic operators.

Chapter 7 Genetic Operator Schemes

Introduction

In the previous chapter we have investigated aspects of the reproduction or duplication step of the reproductive plan. In this chapter we shall turn to the recombination step in which new individuals or devices are produced from duplicates of selected parents using various genetic-like operators.

Our investigation will consist of two phases. First, we will briefly examine the role of the operators when their application rate remains constant throughout evolution. Then we will proceed to develop a scheme through which the application level of each operator will be influenced by the plan's performance. Such a scheme will relieve the researcher of the type of work involved in the first phase of our investigation.

7.1 Operator Probability Settings

In this section we will try to get a feeling for the role of various genetic operators in influencing efficient and effective adaptation. This investigation will by no means be exhaustive for a number of reasons. First, it simply is not feasible, condidering the high cost of our runs, to operate multiple experiments to test the effect of different parameter settings for our operators.

Secondly, it is very doubtful that such experiments would produce many significant differences. As we discovered in our initial artificial task, reproductive plans are generally insensitive to changes in operator probability settings except in extreme cases. However, it was important to have operators which played different roles and allowed steps of diff-

erent sizes to be taken in the search space. We also expect that there will be a fair amount of interaction between various operators. Therefore, it would be difficult to obtain information about the optimal setting of a particular operator without considering a wide range of settings for all the other operators.

Third, even if we did find some optimal probability settings, we still would not have greatly advanced our knowledge about reproductive plans. These settings would obviously depend on the specific operators used and even more so on the particular task, chromosome size, population size, etc. One cannot be expected to go through this optimization procedure for every new operator, task, or chromosomal representation.

What information can we expect to get from this first phase of investigation on genetic operators? First, we can obtain an initial estimate of the role of various operators. For example, we would like to know if the different mutation operators actually have different effects, or if crossover and inversion really help the process of adaptation. Secondly, by investigating the effects of operators at different stages of adaptation we can re-evaluate the need for a scheme which modifies probabilities during adaptation. As one may recall, our work with the initial artificial task demonstrated such a need. Finally, our experiments with a variety of probability settings will give us a better estimate of the performance of reproductive plans on our tasks. In the previous chapter we used the same parameter values for all experiments.

Now let us look at some of the more important experiments. As we mentioned earlier, the crossover and inversion operators are applied with certain probabilities. However, since there are usually about 110 genes per chromosome, the mutation operators may be applied a number of

times, this number being chosen from a given distribution. The distribution is specified by a single parameter designating the maximum possible number of mutations. The distribution is a combination of a uniform distribution plus a possible weight at zero. Figure 7.1.1 shows some possible distributions along with their identifying parameters.

Table 7.1.1 summarizes the first few experiments using a 12/6 population on the difficult task. Experiments 1 and 2 clearly indicate that operating alone, mutation 1 is significantly superior to mutation 2 both after 50 and 100 generations. One may recall that mutation 1 randomly changes a single point in an n-tuple while mutation 2 randomly replaces the whole n-tuple. Experiment 3 used both mutation operators; performance after 50 generations was similar to experiment 1, but improved little in the next 50 generations. This could be directly attributed to the inability to take small enough steps later on in the search process. These results seem to indicate that mutation 1 is a much more valuable operator than mutation 2. Experiment 4 with the introduction of crossover and inversion proved to be significantly superior to all the others after 50 generations. However, at the end of 100 generations, experiment 1 had caught up. Again this can be attributed to the inability of experiment 4 to decrease the application rate of its operators and take smaller steps in the search space.

Figure 7.1.2 displays the adaptation curves of the population mean for typical runs of experiments 1, 2 and 4. Experiment 4 is characterized by large sudden gains and then a leveling off. This phenomenon is probably due to the crossover operator which results in large changes in performance more often than the mutation operators which in-

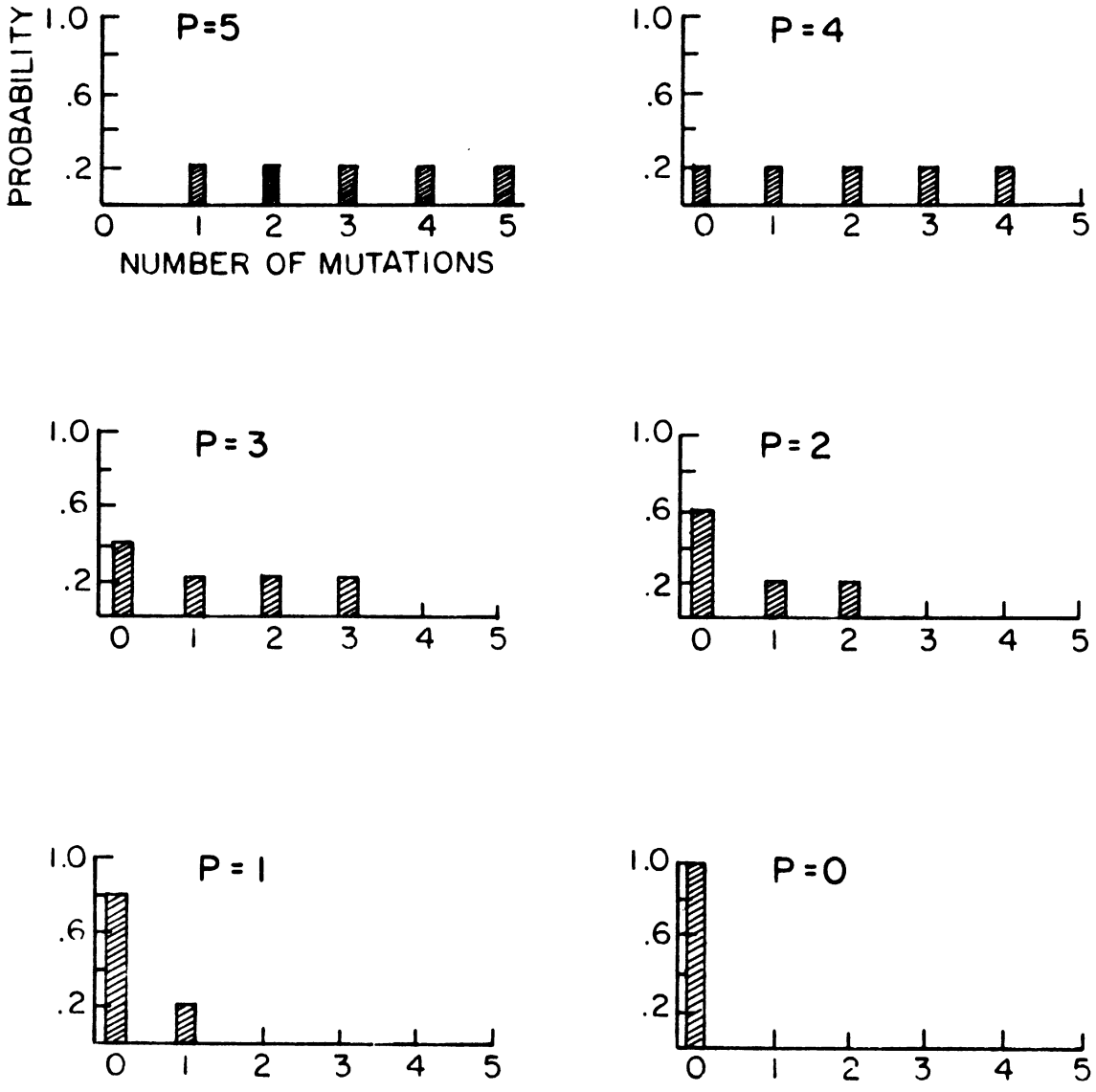


Figure 7.1.1 Possible mutation distributions.
P stands for parameter value.

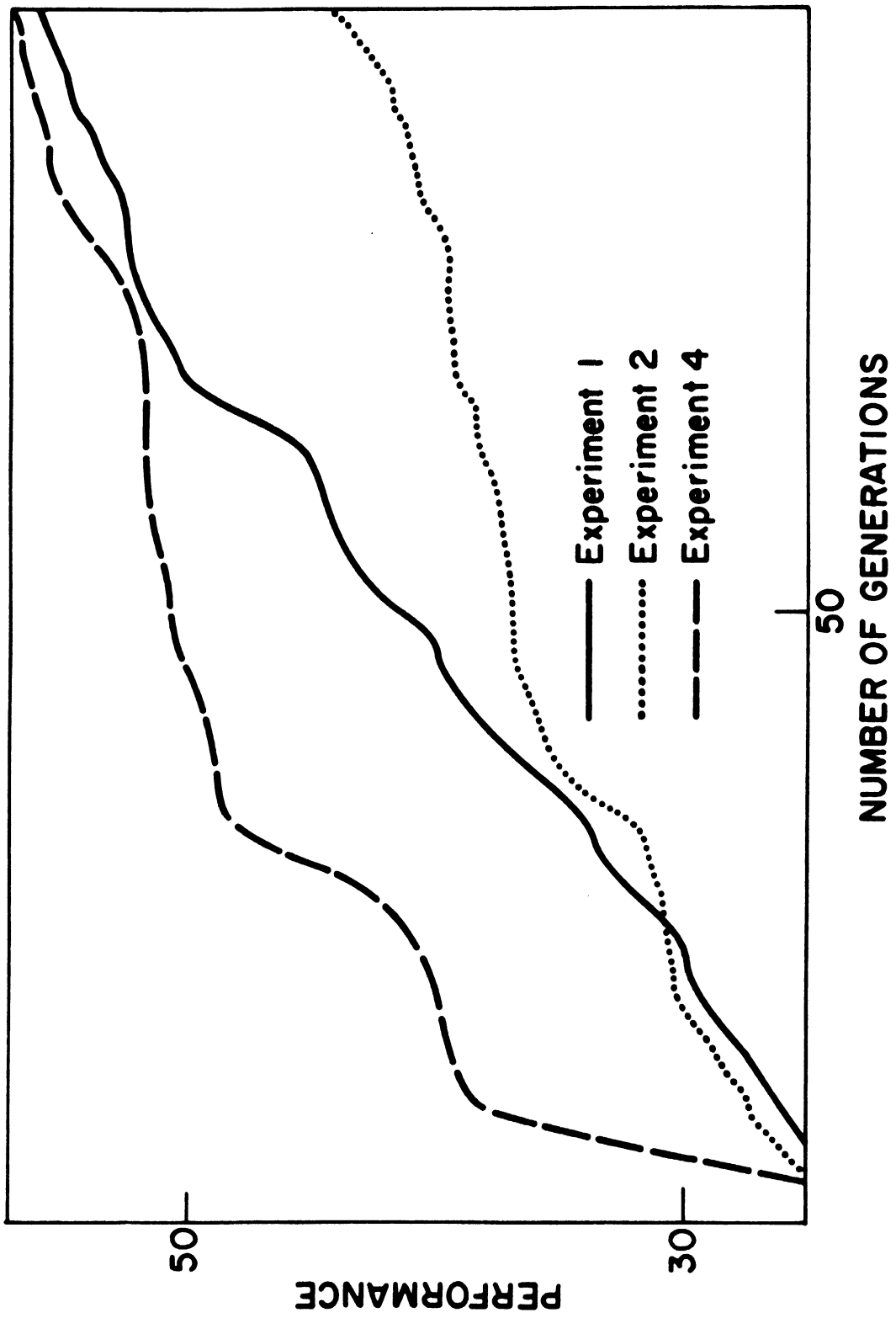


Figure 7.1.2 Adaptation curves for typical runs of selected experiments

EX NO	POP SIZE	TASK	STAT. CRS	PARAMETERS INV	M1	M2	GEN 50	GEN 100	SIGN
1	12/6	D	.00	.00	5	0	44.9	56.4	*
2	12/6	D	.00	.00	0	5	38.4	47.2	*
3	12/6	D	.00	.00	5	5	45.0	48.0	
4	12/6	D	.50	.25	5	5	51.1	57.1	*

Table 7.1.1 Initial Experiments on Genetic Operators

duce a more gradual performance curve. Figure 7.1.2 also reinforces our practice of using a number of stopping rules for the maximum utility criterion. Using a generation stopping rule of less than 50 generations, experiment 4 is superior with little difference between 1 and 2. A generation stopping rule after 50 generations shows experiment 2 inferior with little difference between 1 and 4. However, a slope stopping rule in this case would be inadequate since experiment 4 might get shut off around generation 60 while experiment 1 would continue past generation 100. Such a disparity in the number of samples would prohibit a just comparison. Therefore, we will continue to use at least two stopping rules for future comparisons.

A similar set of experiments run on the easy task yielded results along the lines of those displayed in Table 7.1.1. Table 7.1.2 presents some results using a 20/10 population on the easy task. The main purpose of these experiments was to test the effects of the level of the crossover and inversion operators versus the level of the mutation operators. One general observation is the relative insensitivity of performance despite the rather large range of operator probabilities used.

More specifically, it was interesting to see that experiment 7 with very high crossover and inversion probabilities and no mutation, per-

EX NO	POP SIZE	TASK	CRS	INV	M1	M2	GEN 35	STOP MEAN	STOP GEN	SIGN
5	20/10	E	.2	.2	5	5	76.2	78.0	56	*
6	20/10	E	.5	.5	5	5	78.6	83.2	70	
7	20/10	E	1.0	.9	0	0	80.9	83.8	60	*
8	20/10	E	1.0	.9	3	3	80.7	83.4	61	
9	20/10	E	1.0	.9	10	10	77.7	80.0	51	*

Table 7.1.2 Summary of Experiments

formed as well as any other experiment. Obviously there was sufficient genetic material within the 20/10 population to permit adaptation without mutation. However, a similar experiment using a 12/6 population was not as successful; moreover, experiment 7 without inversion produced virtually no adaptation whatsoever. This suggests some strong interdependencies involved in the optimal operation of reproductive plans. One must use inversion and a relatively large member population for crossover to work well. Under these circumstances crossover works very well, significantly outperforming experiment 5 with low crossover and high mutation rates. Experiment 7 also significantly outperformed experiment 9 which utilized all operators at very high levels. The other experiments were not significantly different from experiment 7.

At this point we decided to introduce two additional operators. Mutation 2 had not proved to be a very valuable operator. This leaves mutation 1 which induces an extremely local search and crossover which takes very large steps in the search space. We felt that there should be some operators which could take some medium-sized steps in the space.

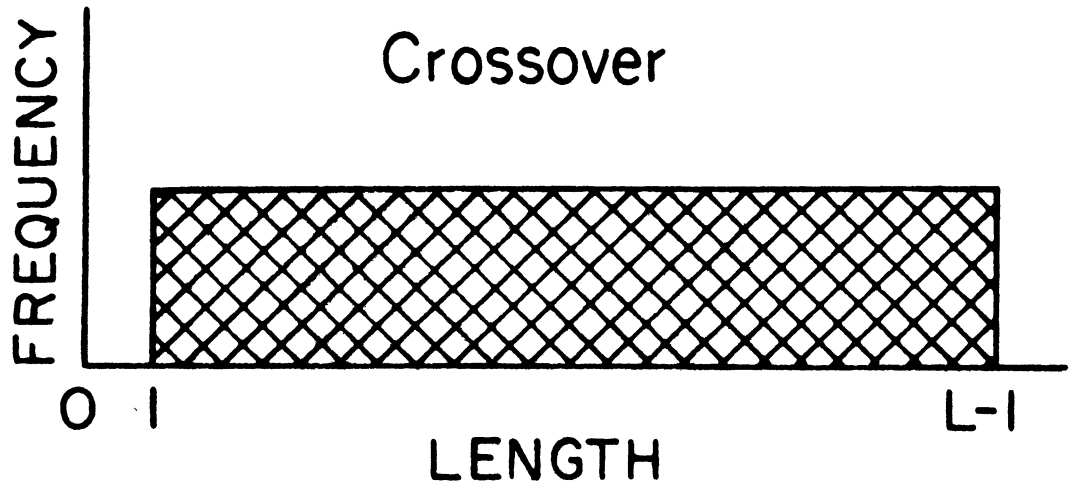
Double crossover was a readily available operator which is similar to crossover except that two breaks occur and the center portions of

the strings are exchanged. On the average, double crossover results in the exchange of segments which are shorter than those exchanged using crossover. Figure 7.1.3 shows frequency distributions for the length of exchanged segments for crossover and double crossover operating on strings of length L.

Crossover and double crossover will *not* be applied independent of each other. At most one will operate on each pair of individuals. The sum of the "probabilities" of both operators will be used to determine probabilistically if either operator will be used. This sum will never exceed one. If one of the operators is to be applied, then the individual probabilities are used as weights to determine which crossover operator will be used.

A new mutation-like operator, called mutation 3, will modify two adjacent n-tuples simultaneously. If the two detectors designate a total of six or more mesh points, then they are transformed into three smaller detectors by regrouping the same mesh points. If the two adjacent detectors specify a total of five or less mesh points, then they are combined to form one detector using all the points. This new operator permits changes in detector composition and size without changing the actual mesh points investigated. Both of these new operators fall between crossover and mutation 1 in terms of the step size taken in the search space.

The new operators seemed to have a marked effect on the reproductive plan's performance. Table 7.1.3 summarizes an interesting set of experiments operating with a 12/6 population on the difficult task. For experiment 11, double crossover was used instead of crossover and mutation 3 was used instead of mutation 2. The frequency of mutation 3



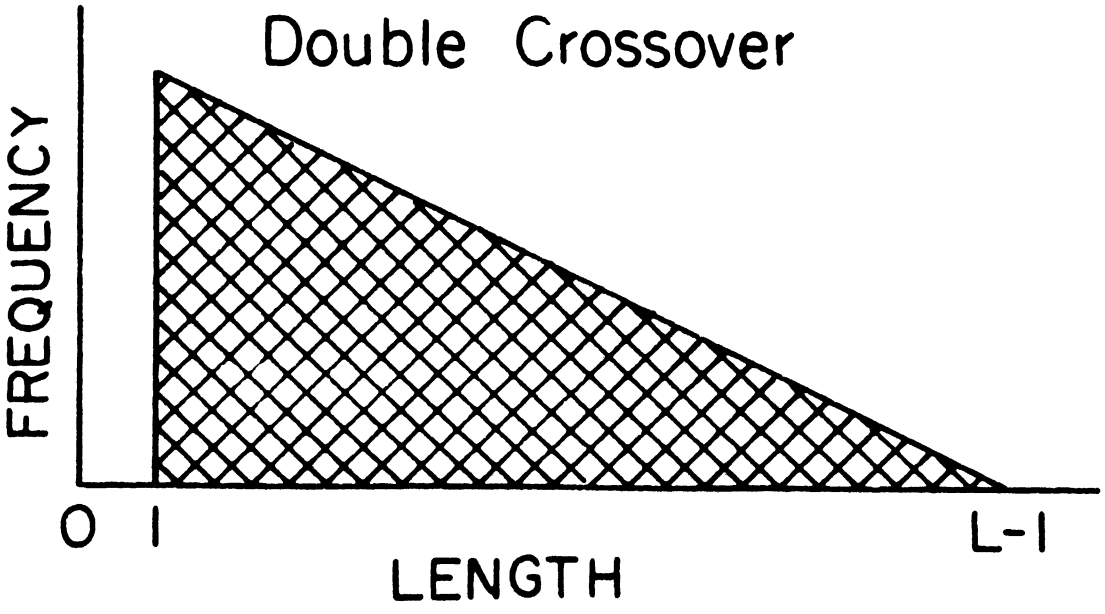


Figure 7.1.3 Length of exchanged segments for crossover and double crossover

EX NO	CRS	DCR	INV	M1	M2	M3	GEN 50	GEN 100	GEN 200	GEN 300	SIGN
10	.5	0.0	.5	5	5	0	49.2	60.0	62.1	65.5	
11	0.0	.5	.5	5	0	3	54.1	65.9	71.9	75.8	*

Table 7.1.3 Some Effects of Double Crossover and Mutation 3

was less than that of mutation 2 since mutation 3 affects two detectors upon each application. Experiment 11 produced significantly superior results (at the 2% level) for every test after generation 50, demonstrating that this setting was both more efficient and more effective. A comparison using the slope stopping rule was not possible due to a large difference in the number of generations elapsed before stopping.

A number of other experiments were run with these new operators. Table 7.1.4 summarizes experiments using a 12/6 population on the easy task. Only experiment 14 using mutation 3 alone proved to be significantly inferior. Again comparisons using the slope stopping rule were not possible due to the large differences in the stopping generations. However, experiment 12 again points out the need to have probabilities vary through evolution. After 35 generations experiment 12 performed as well as experiments 13 and 15. However, experiment 12 terminated much earlier than the others presumably due to its use of too many operators at moderate to high application levels. Performance improved when some of the operator parameters were set to zero, although it was not extremely important which parameters these were.

The need for low probabilities at later stages of evolution was even more evident upon closer examination of new population members. A *new population member* is a newly-formed individual which performs

EX NO	CRS	DCR	INV	M1	M2	M3	GEN 35	STOP MEAN	STOP GEN	SIGN
12	.3	.3	.5	5	5	3	74.2	75.8	49.3	
13	0.0	.5	.5	5	0	3	77.3	82.4	67.5	
14	0.0	0.0	0.0	0	0	4	65.8	67.1	53.3	*
15	0.0	.5	.5	5	0	0	75.3	85.4	99.8	

Table 7.1.4 Additional Experiments with Double Crossover and Mutation 3

well enough to take the place of a parent in the current population. A tabulation was maintained on every new individual to determine how it was created (i.e., via how many mutations of each type, via crossover or double crossover and/or inversion). In early generations the usage of various operators in new population members was consistent with the distributions which generated the operator frequencies. In other words, there was no selection for individuals formed via a certain number of mutations or via crossover. However, in later generations there was a definite selection for individuals with fewer changes, e.g., a lower number of mutations. Figure 7.1.4 shows the observed frequency distribution of the mutation 1 operator among new population members during the last 10 generations of a particular run. The generating distribution (for all new individuals) was uniform in this inverval showing a definite selection for individuals with few mutations.

This evidence seems to solidify our presemption that we need a scheme which can modify the parameters of our operators as the needs of the plan dictate. Real chromosomes actually have sites among their genes which regulate mutation rates [Helling, 1968]. Although a self-regulating system may not always perform as well as one in which optimal

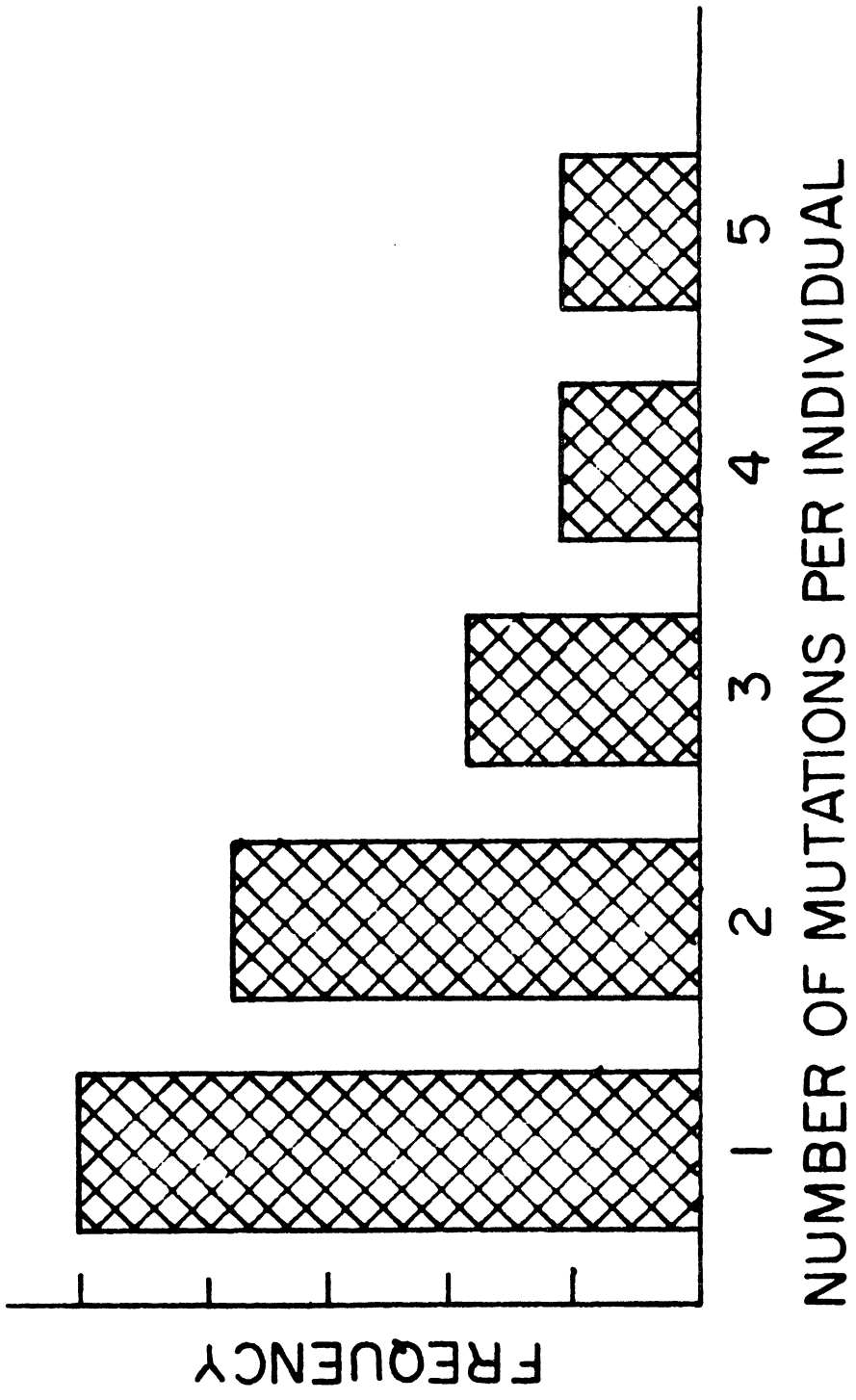


Figure 7.1.4 Mutation 1 frequency in new population members

settings have been determined (usually after extensive studies), the self-regulating system certainly would be more flexible and would relieve the researcher of much of the work we have performed in this section.

Summary

The work of this section has proved to be very successful in accomplishing its goal. First, we have gained a good estimate of the worth of the various genetic operators and their roles. Mutation 1 seemed to be the most valuable mutation operator, with mutation 2 the least valuable. Furthermore, all the mutation operators seem to play more important roles later in the evolutionary process when small step sizes are beneficial.

The crossover operators coupled with inversion also play a valuable role especially in regard to the efficient operation of reproductive plans. They are capable of providing for significant adaptation even without any of the mutation operators. However, the optimal use of crossover and inversion seems to demand a population of a size capable of maintaining a rich mixture of genetic material.

As a by-product of these experiments we have established some "good" overall parameter settings and performance levels which will serve as optimal guidelines for future comparisons. Furthermore, we have established that our reproductive plans are relatively insensitive to changes in the operator parameters. This is probably due to the fact that different operators play similar roles in searching the space. However, large changes in parameter settings (eliminating an operator or creating a new one) can have substantial effects in the plan's efficiency and effectiveness. This is particularly true when only a few operators

are being used.

We have also gained some more insight into the use of stopping rules for the maximum utility criterion. Comparisons at different generations proved very valuable in identifying good and bad aspects of various settings. On the other hand, the slope stopping rule usually did not result in an equitable comparison in terms of performance levels. However, by comparing the number of generations before stopping, we did gain valuable information about settings which were not capable of sustaining sufficient adaptation later in evolution. This information will prove to be very helpful in the next section on parameter modification schemes.

Finally, we have now massed enough information to justify our belief that a parameter self-modification scheme would be extremely beneficial. We have observed distinct differences in effective operator application rates at different points in the evolutionary process. In general, most operators should be applied less frequently in the later stages of evolution. However, in such a situation (low parameter settings) the plan becomes more sensitive to particular operators. This sensitivity should be used to maintain the valuable operators and effectively turn off the poorer ones. We shall now turn to the task of developing a suitable parameter-modification scheme.

7.2 Parameter Modification Schemes

Introduction

Our immediate goal in this section is to develop a scheme which can suitably adjust and modify the genetic operator parameters as the needs of the plan dictate. To reiterate, a scheme of this type will serve two major purposes. First of all, it will make reproductive plans more general and flexible since it will facilitate their application to new tasks using various chromosomal representations. In this case the researcher can be fairly certain that a few runs will be representative of the plan's performance using a certain set of operators, since the plan will not be biased by a predetermined set of operator parameters. Furthermore, after observing the plan's use of the operators he may decide to eliminate some and create new ones similar to ones which have proved to be effective. This would be a more interesting and valuable endeavor than just testing parameter values for a given set of operators.

In addition to increasing the flexibility of reproductive plans, the parameter modification scheme should in the long run make them more efficient and effective. The ability to adjust parameter levels at different stages of evolution should allow a longer period of significant adaptation. Furthermore, the plan should be able to turn up or down the application level of individual operators according to their worth in the particular task environment.

Examining a parameter modification scheme within our formal framework for adaptive systems, we can view it as sort of a meta-plan within the reproductive plan. In this meta-system the set $\mathcal{A}$ is the set of all parameter value combinations for a given set of operators. The en-

vironment determines how well a particular parameter setting performs, i.e., how effective this parameter setting is in generating a high proportion of good offspring. As we have seen, the environment in this meta-system changes since a genetic operator (and therefore its application level) has varying worth at different stages of adaptation. The different schemes that we will develop in this section will be taken from the set of plans $\mathcal{P}$ in this system. As opposed to reproductive plans, these plans will not be first-order but will use additional feedback from the meta-environment. This mode of operation stems from the difficulty of obtaining an accurate estimate of the worth of one parameter setting. Such an estimate is only possible after the setting has been used many times, but this involves running entire experiments as we did in the previous section.

Let us be a bit more explicit about how we will extract information which will enable us to modify parameter settings. Presumably at each point in evolution there is an optimal setting which will be most successful in generating valuable new offspring. Suppose the plan is using a parameter setting which results in higher application rates of the operators than would be dictated by a better setting. In this case the best offspring will on the average result from lower application levels. Since a parameter setting effectively defines a distribution of application levels, offspring resulting from low application levels are possible. Examination of operator usage in creating the better offspring, i.e., those which have become new population members, should indicate that low application rates are favorable and the appropriate distribution parameters will be lowered slightly.

The situation here is very similar to that of Samuel's checker player. We must extract and use as much information as possible so as to respond quickly and validly to changes in the environment. In order to do this we must make assumptions such as independence of components. This certainly is not an accurate assumption since we have reason to believe that the components (i.e., the genetic operators) are not independent of one another.

However, we can justify this decision. We do not need an optimal parameter setting but only a fairly good one. Therefore, a good approximation which converges quickly is preferable to an optimal solution which takes more time. In other words, the ability to change a setting during adaptation is more important than the ability to find the best setting at any particular point in time since there are probably many good settings, judging from our work in the previous chapter.

Referring again to Samuel's work, one may suggest that a signature table approach would be *a propos*. Certain parameter value combinations could be rewarded as a whole, resulting in a more accurate utility measure for any given complete set of parameter values. However, the problem of accurately filling in the entire table would be even more difficult than in Samuel's case. We can reward a set of parameter values only when they are used to produce new offspring. Since we typically generate 6 offspring per generation, we may expect to have 300 "tests" of different parameter settings after 50 generations. This may be considered a reasonable number for a table with a total of 6 parameters, each parameter quantized to about 5 values. However, one "test" of a parameter setting consists of one sample from the distribution which this parameter setting defines. So we would need at least a few tests

of each setting. Furthermore, the data gathered during the first 50 generations would probably not be applicable to the next 50 generations. In fact, there may be a negative correlation since high parameter values seem to work well during the first phases of adaptation whereas low settings work well later on. We may, therefore, conclude that a signature table approach would not be very suitable.

Criteria for evaluating our meta-plans or probability modification schemes will be different from the usual criterion for evaluating reproductive plans. Certainly the overall performance of the reproductive plan using the parameter modification scheme will eventually serve as a crucial determiner of a scheme's success or failure. However, in developing suitable schemes we shall use some intermediate criteria to guide our work.

A very important criterion will be based on intuition and past experience with reproductive plans. We have developed general ideas for what constitutes favorable parameter settings. For example, a low application level for the genetic operators will generally be favorable in the later phases of adaptation, whereas higher levels are beneficial in the earlier phases. We shall expect good schemes to follow such general guide lines.

Other intermediate criteria will be based upon convergence and stability. By *stability* we mean that the parameter values should vary gradually throughout adaptation without rapid fluctuations. This is a criterion used by Samuel in evaluating his Δ-plans for modifying his linear polynomial. We will say that a scheme *converges* if the parameter settings for different *runs* at similar points in evolution fall "close" to each other. It is questionable whether we should expect a scheme to

converge. Although each run starts out with the same parameter setting, they could all pass through and end up with very different settings which were equally good. However, if a scheme did converge in the above sense, we would have some assurance that the scheme was not overly sensitive to the random aspects of reproductive plans and did actually extract relevant information.

In the final analysis of parameter modification schemes, we must resort to the maximum utility criterion to evaluate the reproductive plan which uses the schemes. In this case, we will compare plans with and without the parameter modification scheme operating on our previously run task and also using some different sized chromosomes to change the demand for different operators. These latter experiments should test the ability of the parameter modification scheme to operate in a situation different from that in which it was developed. This is very important since a major purpose for a parameter modification scheme is to give reproductive plans the added flexibility to operate efficiently on new tasks with different chromosomal representations and genetic operators. In addition to the above experiments, all of which will start with the same parameter values, we shall compare the performance of a plan with the parameter modification scheme to a variety of plans without the modification scheme but which use different parameter settings. This should indicate whether one may justifiably dispense with the type of experiments we performed in the previous section without greatly sacrificing the effectiveness of the reproductive plan. A good plan with a parameter modification scheme will obviously be more efficient since we can then eliminate all the experiments in the previous section.

Suitable Modification Schemes

One approach to parameter modification is similar to that actually observed in nature. The parameter values would be encoded into the individual strings and would determine how their offspring would be formed. The offspring could assume the parent parameter with small modifications or in the case of two parents, some average of the parent parameters. However, this scheme is similar to the signature table approach and subject to similar criticisms. An offspring's parameters are determined from his parents', yet the actual operator frequency which determined the offspring is just one point in the distribution induced by the parrents' parameters. Furthermore, we would have to worry about loss of variation in the meta-population of parameter settings. Considering the relatively small population size and the small number of generations over which evolution takes place, there would probably be an insensitivity to appropriate modifications.

Therefore, we have devised an alternate scheme which hopefully will alleviate the sampling problem somewhat. The scheme works in the following manner. One set of parameter values is maintained for all individuals. During each generation a tally is kept to determine how many times each operator is applied and is successful in contributing to the creation of a new population member. The reader should recall that a *new population member* is a new individual which performs well enough to take the place of a parent in the current population. After a certain number of new population members are observed, the parameter values are modified to an extent determined by the observed frequencies.

If P(t) is some parameter value at generation t, and O(t+n) is the observed value for the parameter *in new population members* between gen-

erations t and t+n, then

$$P(t+n) = P(t) + [O(t+n) - P(t)]\ A$$

where A is some expression, not necessarily constant, which controls the amount of change taking place ($0 \leq A \leq 1$). The actual form of A will depend upon whether a crossover parameter or a mutation parameter is being modified. A may also vary over the different schemes that will be tried.

For the crossover operators P(t) is the probability of crossover while O(t+n) is the observed relative frequency in new population members. For the mutation operators, P(t) is the identifying parameter of a generating frequency distribution while O(t+n) is the parameter of a distribution whose mean matches the observed average frequency of the particular mutation operator in new population members. If P(t) falls between integers, we combine the two integer distributions in a weighted manner.

Our immediate goal is to determine :

1) the best form of A,
2) the number of individuals observed before a change in probability settings is made,
3) Appropriate *levels* of starting probabilities, i.e., high, medium, or low,
4) what other forms of control might assist this general parameter modification scheme.

Inversion was not put under direct selection since it does not directly affect an individual's performance. Instead, its value is set equal to the average value of crossover and double crossover.

In the following discussion we shall explain the details of each

scheme, briefly describe some results and then motivate the development of the next scheme.

Scheme 1

Definitions:

$$PGAP = \begin{cases} 1-P(t) & \text{if } O(t+n) \geq P(t) \\ P(t)-.1 & \text{if } O(t+n) < P(t) \end{cases}$$ (defined only for the crossover operators)

NTOT = the number of new population members observed between successive parameter changes, i.e., between t and t+n.

NEW = the number of possible new population members per generation.

MIN = the minimum number of new population members which must be observed before a change in probabilities is made.

K = a nonnegative constant.

Modification takes place after at least MIN new population members have been observed and after a generation is completed. Therefore, $MIN \leq NTOT < MIN + NEW$. The PGAP factor will determine the amount of change possible for the crossover operators, the minimum possible crossover probability being .1. Such a minimum value is needed so that an operator will not be turned off and thereby eliminated from future usage. An alternative method would be to continually introduce random application of an operator independent of its probability.

The A factor has the following form:

For the crossover operators,

$$A = \frac{(PGAP)(NTOT)}{(NEW+MIN+K)}$$

For the mutation operators,

$$A = \frac{(NTOT)}{(NEW+MIN+K)}$$

The purpose of the PGAP factor is to keep the crossover operators from falling below the .1 level while also decreasing changes in values near the extremes. We did not feel that a similar factor was needed for the mutation operators. The remaining factor insures that A remains at a suitable level below 1.0. MIN was a constant for this scheme.

The goal of the initial experiments was to determine good values for MIN, K and the starting parameters. A low value for MIN and a high value for K means that changes in values would be made very often but by small amounts. This did not prove successful since the changes were based upon very small samples, resulting in too many random modifications. The initial probabilities changed very little throughout evolution. On the other hand, a high value for MIN and low K was even worse. Very few changes were made and when they were, averaging had taken place for so long that the observed rates closely approximated the actual probability parameters. Fairly good results occurred when MIN = NEW and K = 5. Making MIN dependent upon NEW has the virtue that the results are less dependent upon the number of offspring generated per generation. Therefore, MIN will be larger for the 20/10 population than for the 12/6 population.

Certain failings of this first scheme became evident immediately. The expression,

$$\frac{NTOT}{NEW+MIN}$$

was present to insure that $A < 1$ even if K = 0. However, NTOT was gen-

erally closer to MIN than MIN + NEW especially in later generations. Therefore, with MIN = NEW, A = 1/2. Yet in earlier generations, A was often near 1. This fluctuation in A dependent upon how many new population members were created during the generation that NTOT exceeded MIN was not desirable. Furthermore, this effect was difficult to analyze with the term K in the denominator.

Other problems also existed. In late generations there were fewer new population members. Therefore parameter changes occurred less often just when they should have been changing a great deal, hopefully decreasing. As a result, high initial parameter values tended to remain fairly high throughout evolution.* The lack of a lower limit on mutation also caused a problem, since some rates became so low that the operators were effectively shut off. This was particularly serious when initial parameters were low.

Scheme 2

Scheme 2 was designed to correct some of the failings of Scheme 1. First MIN was allowed to change values during evolution. As soon as the marginal gain in performance during the previous 15 generations became less than 3.0, MIN was halved and remained that way for the rest of the run. This was to provide for more changes in parameters during later generations when new population members were infrequent. Hopefully, this change would induce a reduction in most operator rates so that smaller steps could be taken in the sample space.

---

*By "high initial parameter values" we mean crossover rates of .5 and mutation distributions with upper limits of 5. Low values are crossover rates of .2 and mutation distributions with upper limits of 2.

The A factor was changed as follows:

For the crossover operators,

$$A = \frac{(PGAP)(NTOT)}{MIN+.3(NEW)}$$

with $0 \leq PGAP \leq .5$.

For the mutation operators,

$$A = \frac{K(PGAP')(NTOT)}{MIN+.3(NEW)}$$

where,

$$PGAP' = \begin{cases} P(t)-1.0 & \text{if } P(t) \leq 2 \text{ and } 0(t+n) < P(t) \\ 1.0 & \text{otherwise} \end{cases}$$

and K is a constant such that $0 \leq K \leq .5$

The factor $\frac{NTOT}{MIN+.3(NEW)}$ was decided upon after inspection of values of NTOT. It eliminates some of the wide variations noted above but still enables a large NTOT to carry more weight than a smaller one when changes are made. Although this factor may possibly be greater than one, the limits on PGAP and K insure that A will be less than one.

The restriction on PGAP was necessary since crossover probabilities underwent large fluctuations especially when P(t) was low and 0(t+n) was very high. 0(t+n) may be very high in populations which lose their variance so that many individuals resemble others. In this case a crossover may produce no new mixing but the resulting offspring may prove to be good due to the action of other operators. The PGAP' factor provides for a minimum mutation value of 1.0 and reduces change in mutation rates when they are already low and still decreasing.

The results with this scheme were mixed. With 20/10 populations the results were uniformly good, independent of starting probability levels and modification in MIN and K (K was set at .3 or .5, MIN set at NEW or .5(NEW)). On the other hand, the 12/6 populations exhibited

erratic behavior between runs. Low starting parameter values resulted in inferior final payoffs; large populations did better with low starting values. With high starting values the small populations showed no clear cut results, performing well during some runs and poorly during others. In addition, the small populations were not successful in reducing high parameter values near the end of evolution. The large populations were more successful at this.

These observations would support a hypothesis that large populations are certainly superior to smaller ones and worth the extra cost involved. Certainly, one explanation is due to the larger sampling rate of the large population. Not only do the large populations have more chances to create superior individuals, but in general they also have more information available to make better adjustments in their parameter settings. The sampling problem becomes more crucial with parameter modification schemes, since not all sampled (newly created) individuals provide data samples to the parameter modification scheme; only those which become new population members are used.*

There is an additional cause for the inferior results of the small population. When probability settings are very low, it becomes very possible than an offspring will not be acted upon by any operator and therefore remain identical to its parent and obtain the same payoff. This serves only to reduce the variance within the population and obviously hurts small populations more than large populations.

---

*One might argue that keeping a record of rates of the bad individuals might prove useful. However, it is not clear how this information could beused along with the current information, to modify parameters. Furthermore, in late generations, most new individuals are bad due to peak effects. Using such information might just add a noise effect to the information obtained from new population members.

Scheme 3

Scheme 3 results from a few small changes in Scheme 2. First of all, each new individual is checked to make sure it is sufficiently different from its parent. An inversion and/or one mutation 1 was not considered a sufficient change; any greater chage was sufficient. If an individual was not sufficiently different, additional operators were applied.

The second modification involved final parameter values which still seemed too high in many cases. Therefore, every generation that the marginal gain for the last 15 generations was less than 3.0, each mutation rate was decreased one-half the distance to 2.00 (given that it was greater than 2.00) and each crossover probability was decreased one-half the distance to .1. This was in addition to reducing MIN.

These modifications seemed satisfactory. In particular, the small populations exhibited better performance than with Scheme 2. In addition, performance levels were approaching the results achieved with the best parameter settings when no parameter modification scheme was used.

Scheme 4

Scheme 4 again involves minor modifications prompted by observation. In this case the MIN reduction lasts only while the marginal gain is less than 3.0; if it rises above 3.0 later on, then MIN resumes its initial value. Such a measure was deemed necessary since an early leveling off followed by a resumption of a normal increase may result in a low value of MIN to persist when it is not needed.

Similarly decreasing mutation and crossover rates every generation

that the marginal gain was less than 3.0 seemed to result in lower rates than desired and lack of flexibility, especially for the crossover operators. Therefore, the reduction process was applied only as the marginal gain *fell below* the 3.0 point. A few runs with this scheme indicated that the unwanted phenomena had been eliminated.

Quantitative Results: Parameter Values

At this point we tried to get a more quantitative measure of the value of the parameter modification scheme. A number of experiments were run using 12/6, 20/10 and 20/14 populations. The introduction of the 20/14 population at this point seemed very fitting. The 20/14 population costs no more timewise than the 12/6 population since it still samples only 6 individuals per generation. On the other hand, it has a larger population base (14 individuals) than the 20/10 population making it less susceptible to loss of variance. We might note here that the 20/14 population would have provided no benefit with *selection schemes* 1 and 2 since the 6 offspring would always be apportioned to the top 6 or less population members. The rest of the population would never produce offspring. However, Selection Scheme 4 quite often distributes excess offspring among the whole population.

The difficult task will be used for all future experiments since many of our experiments are not reaching nearly maximal performance (empirically observed) on the easy task. Operator parameters will be initially set to medium levels (crossover operators at .4, mutation parameters at 4.0), MIN will equal NEW initially, and K will be .5.

Let us first look at the changes in parameter values with respect to the stability and convergence criteria. Figure 7.2.1 shows the changes

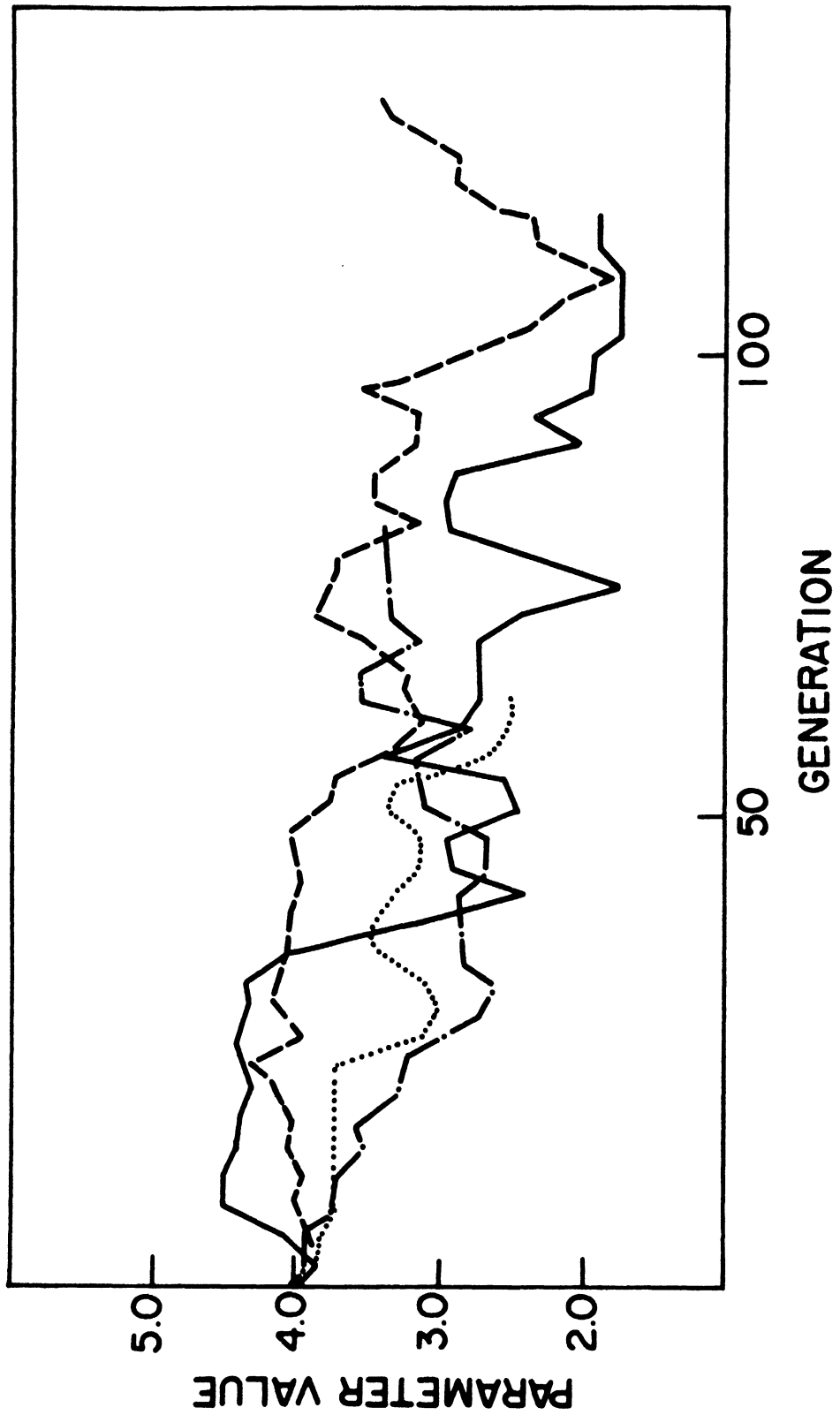


Figure 7.2.1 Change in the mutation 1 parameter for different runs of an experiment

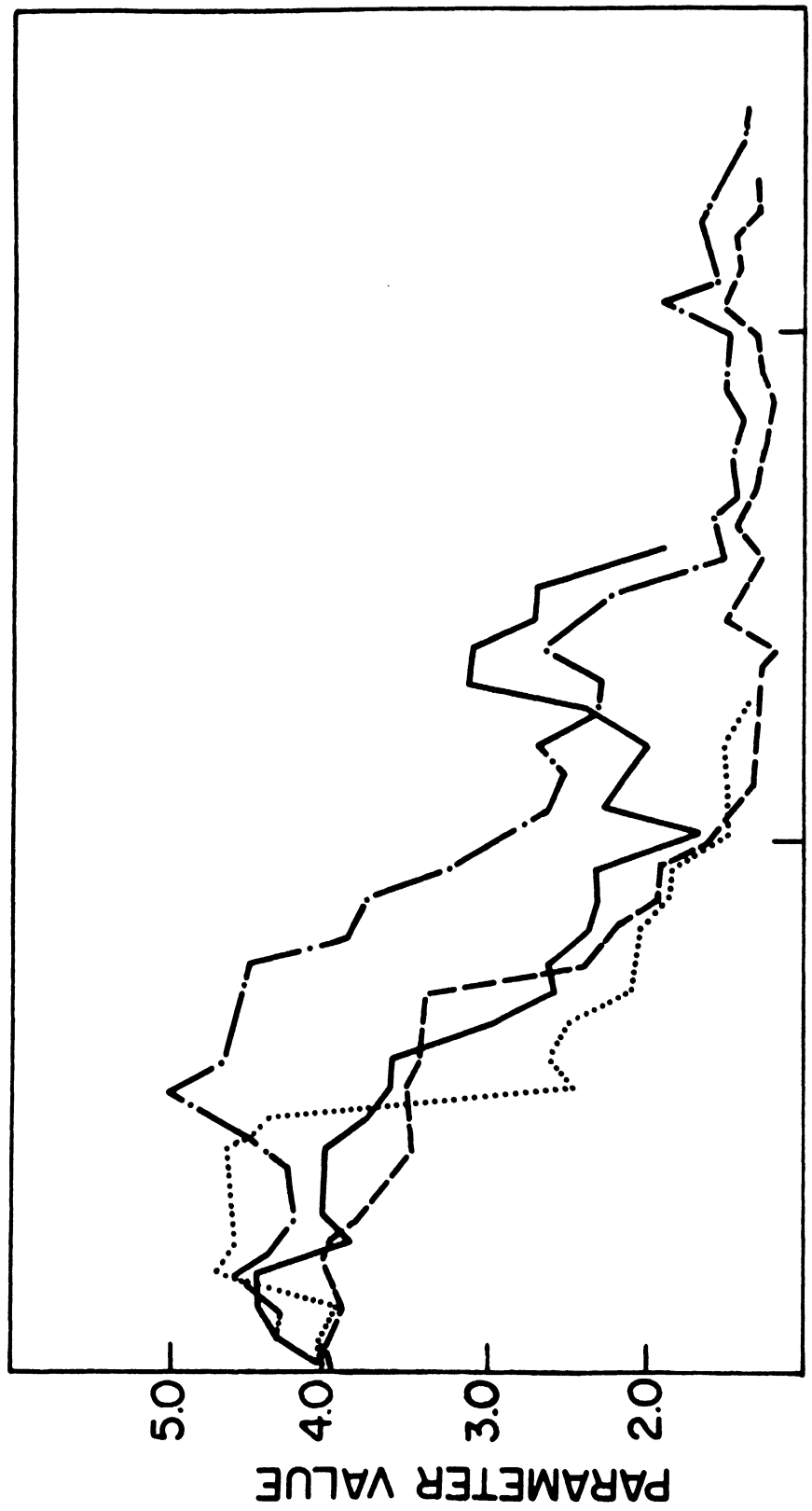


Figure 7.2.2 Change in the mutation 2 parameter for different runs of an experiment

in the parameter value for mutation 1. Each curve represents a different run plotted until the slope stopping rule was exercised. Figure 7.2.2 is a similar graph for mutation 2. The experiment used a 12/6 population. The parameter value for mutation 2 seemed to undergo more gradual changes which were very similar between runs despite the fact that some runs ended much sooner than others. Also the variance in the final values for mutation 2 was smaller than that of mutation 1 for this experiment indicating that mutation 2 converged more than mutation 1.

Another way to test for "convergence" of final parameter values is to determine if the values of different operators converge to significantly different levels. Such a result would indicate that the parameter modification scheme can sense the need for different operators and consistently adjust the parameter values. Figure 7.2.3 shows the changes in all three mutation parameters during two runs. We can observe two distinct phenomena from this figure. During the first third of the generations, mutation 2 and 3 operate at higher levels than mutation 1. This could indicate a selection during this phase for mutations which induce large changes in the composition of the string. Secondly, at the end of each run (using the slope stopping rule) mutation 1 is higher than mutation 2 which is in turn higher than mutation 3. This indicates a selection in later stages of adaptation for mutation operators which make smaller changes.

Let us now try to verify these observations quantitatively. Table 7.2.1 gives the average ending values of the mutation parameters for a number of experiments using different population sizes and different chromosome lengths. The entry under NO. DET indicates the average number of detectors used in the chromosome for each experiment. Under

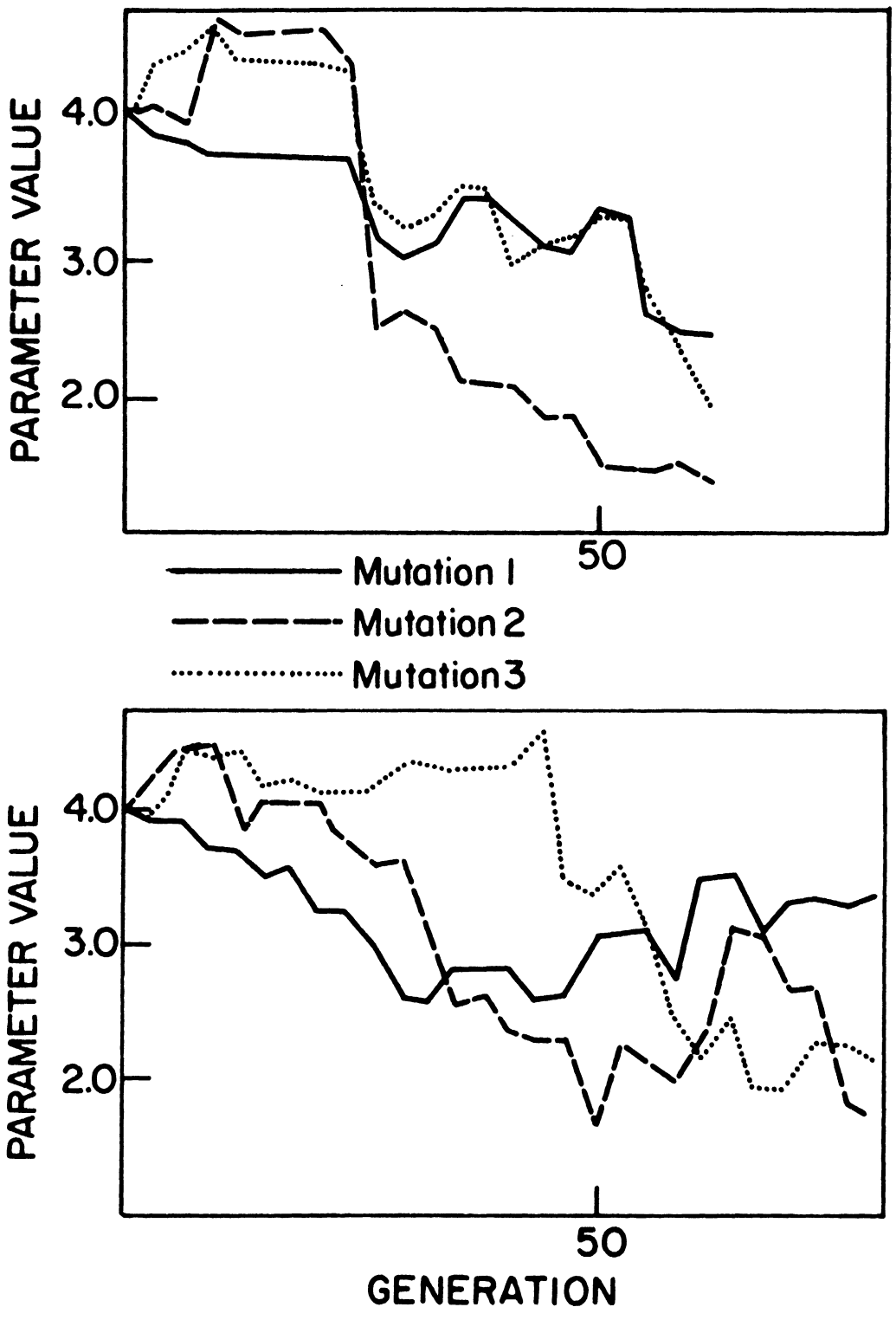


Figure 7.2.3 Comparative changes in mutation parameters for two different runs

the heading SIGN PAIRS we have indicated which pairs of mutation operators ended up at significantly different parameter value levels. All parameter values for all experiments started at 4.0.

As we can see in all experiments mutation 1 ended at a significantly higher level than mutation 2. In addition, mutation 3 was often significantly higher than mutation 2 and significantly *different* from mutation 1. These results clearly indicate the ability of the parameter modification scheme to converge to parameter values which are significantly different even though all values were the same initially. In addition we can see that all values are lower than the initial value.

We can perform some additional important tests using the data in Table 7.2.1. We have indicated above that a good parameter modification scheme should be sensitive to the needs of different tasks in addition to the needs of one task at different generations. Therefore, we shall test to see if one operator ends up at significantly different levels in different experiments. Although we have not really changed the task in the different experiments, we have changed the need for different operators by decreasing the chromosome length or in effect changing $\mathcal{A}$.

The first three experiments use the same chromosome length but differ in the composition of the population. For these experiments there was no significant difference in the final levels for all three mutation operators. This result indicates a convergence to a mutation level independent of population size. Experiments 3-5 use the same population size but differ in chromosome length. For these experiments we find a distinct difference in final parameter levels. For mutation 1 all three experiments ended at significantly different levels, consistent with expectations that the shorter the chromosome, the fewer mutations. This trend was upheld for mutation 2. Mutation 3 was more in-

EX NO	NO. DET	POP SIZE	MUTATION VALUES M1	M2	M3	SIGN	PAIRS
1	100	12/6	2.64	1.33	1.76	1&2,	1&3, 2&3
2	100	20/10	2.23	1.25	2.40	1&2,	2&3
3	100	20/14	3.26	1.46	1.85	1&2,	1&3
4	35	20/14	2.08	1.27	2.80	1&2,	1&3, 2&3
5	17	20/14	1.42	1.10	2.06	1&2	

Table 7.2.1 Ending Mutation Levels

teresting. Only between experiments 3 and 4 were the levels significant. In this case, mutation 3 was higher with the shorter chromosome, a result that was unexpected. Furthermore, mutation 3 is significantly higher than mutation 1 in experiment 4, a reversal of the trends in experiments 1 and 2. Additional studies with short chromosomes have indicated that mutation 3 is indeed more valuable than in the long chromosome situation. These results clearly indicate that the parameter modification scheme can adjust parameter levels in a variety of situations and in a manner which is consistent between runs of an experiment.

Another criterion for evaluating a parameter modification scheme concerns its ability to perform well independent of starting probability levels. Therefore, we ran some additional experiments which were identical to experiments 3 and 5 except for starting parameter values for the mutation operators. These results are summarized in Table 7.2.2. Experiments using 100 detectors showed no significant difference in the final values of different experiments. Using 17 detectors, experiment 9 was significantly different from 8 and 5 in the case of mutation 3 only. This, however, is an extreme situation. The use of only 17 detectors and starting mutation parameter levels of 5.0 results in initially mutating over half of the detectors on the average, in addition to the application of one of the crossover operators. Such a high level of mixing

EX NO	NO DET	START VALUE	END VALUES M1	M2	M3	SIGN
6	100	3.0	3.16	1.28	1.95	
3	100	4.0	3.26	1.46	1.85	
7	100	5.0	2.85	1.38	1.96	
8	17	3.0	1.44	1.10	1.40	
5	17	4.0	1.42	1.10	2.06	
9	17	5.0	2.56	1.34	3.61	

Table 7.2.2 The Effect of Starting Parameter Levels on Final Levels

adds too much noise for the parameter modification scheme to operate effectively. Therefore, we may conclude from these experiments that the parameter modification scheme under reasonable operating conditions can eventually adjust its parameters to values which are independent of initial values.

We have not mentioned the crossover operators in the above discussion due to the lack of any consistent trend. The probabilities of these operators fluctuated more rapidly than the mutation rates, especially near the end of adaptation. At this point in evolution, new population members created by mutation are more common than new members created by a crossover operator. Therefore, crossover probabilities would tend to decrease. However, at the same time, we have an increasing number of individuals in the population which differ only at a few genes due to the formation of offspring using only mutation. Thus it becomes more likely that a crossover will exchange identical segments, except for maybe a few genes, producing an effect similar to mutation. This would

tend to increase crossover probabilities. Since sample sizes are very small at this point in evolution, these couteracting effects often produce much fluctuation in the crossover probabilities. We will discuss possible remedies for this situation near the end of this section.

Quantitative Results: Performance Levels

We must now determine if in fact our parameter modification scheme does result in a more effective search. We have already pointed out the flexibility and efficiency of the scheme since it operates independently of initial parameter settings and chromosome length. In terms of performance, however, we would like to show the following:

1) A plan with the parameter modification scheme attains performance levels significantly higher than the average of other plans using different parameter settings but no modification scheme.
2) A plan with the modification scheme and certain starting parameter values attains performance levels significantly higher than a plan which uses the same starting parameter values without modification.

Table 7.2.3 gives results for some initial experiments. The entry under MOD? indicates the use of the parameter modification scheme. All experiments started with medium parameter values (mutation=4.0, crossover

EX NO	POP SIZE	MOD?	STOP MEAN	STOP GEN	STOP MEAN S.D.
10	12/6	No	60.2	79	6.6
11	12/6	Yes	61.4	99	6.0
12	20/10	No	64.6	90	7.9
13	20/10	Yes	70.2	102	3.8

Table 7.2.3 Initial Results Using the Parameter Modification Scheme

=.4). The experiments using the modification scheme did perform significantly better than the average of other plans with the same population size (goal 1 above). However, they were not significantly superior to the other experiments using the medium starting levels throughout. The lack of significance between experiments 12 and 13 was largely due to the large variance of experiment 12. Unfortunately these initial experiments were terminated using the slope stopping rule so that comparisons during later generations were not possible. It is during later generations, however, that the parameter modification scheme should prove superior due to its ability to take smaller steps.

To test this hypothesis we ran a 20/14 population for 250 generations with and without the modification scheme. Table 7.2.4 summarizes the results. Using the slope stopping criterion the modification scheme was superior, but this can be accounted for by the additional number of generations used before stopping. After 100 and 200 generations there was no significant difference although the modification scheme dominated. However, after 250 generations, the modification scheme was clearly superior. In addition, the variance of the modification scheme was significantly lower.

EX NO	MOD?	STOP MEAN	STOP GEN	GEN 100	GEN 200	GEN 250	S.D. 250	SIGN
14	No	64.2	97	64.7	70.7	71.8	2.9	
15	Yes	67.5	120	64.8	72.3	75.5	1.0	*

Table 7.2.4 The Modification Scheme with a 20/14 Population

The next table compares performance levels of experiments using short chromosomes (17 detectors). We have shown above that the modification scheme converges on different settings for this chromosome. Table 7.2.5 verifies that the result was favorable. At every generation the modification scheme significantly dominated.

Next let us examine what effect different starting parameter values will have on final performance levels. We have seen above that there is no effect on final parameter levels but this does not guarantee that performance has not been affected in the transition period. Table 7.2.6 summarizes these results. Using 100 detectors we find that there is no significant difference using the slope stopping criterion. Similarly, using 17 detectors we again find no significant difference using a number of criteria. Experiment 22 seems to be operating better initially but the others catch up very quickly, despite the fact that experiment 23 began with extremely high settings considering the number

EX NO	MOD?	STOP MEAN	STOP GEN	GEN 100	GEN 150	GEN 200	SIGN
16	No	60.3	98	60.2	62.6	64.3	
17	Yes	71.0	122	70.2	72.2	73.2	*

Table 7.2.5 The Modification Scheme Using a Short Chromosome

EX NO	NO. DET	START VALUE	STOP MEAN	STOP GEN	GEN 150	GEN 200
18	100	3.0	63.8	103	--	--
19	100	4.0	67.5	120	--	--
20	100	5.0	65.2	127	--	--
21	17	3.0	67.3	117	69.8	71.4
22	17	4.0	71.0	122	72.2	73.2
23	17	5.0	65.5	118	68.1	72.4

Table 7.2.6 Effects of Different Starting Parameter Values

of detectors.

As final proof of the overall superiority of the parameter modification scheme we took one of the best reproductive plans (which will be developed in the next chapter) and ran it without the parameter modification scheme. The results are presented at this time in Table 7.2.7 to demonstrate the modification scheme's superiority in this important situation.

EX NO	MOD?	STOP MEAN	STOP GEN	GEN 150	GEN 200	SIGN
24	No	71.9	111	72.9	74.1	
25	Yes	75.5	130	76.8	79.2	*

Table 7.2.7 Effect of the Modification Scheme on One of the Best Reproductive Plans

Critique of the Parameter Modification Scheme

Before ending this section we would like to point out some bad aspects of our parameter modification scheme and to suggest some improvements. Although the scheme has proved to be a valuable addition to our reproductive plan, we believe that some parts of it can still be further refined to work even better.

The first problem involves the starting parameter values which must still be set by the experimenter. We have shown that within a certain range the initial values do not significantly affect final performance especially in the long run. However, in extreme cases and in the short run, initial values do seem to have some effect. As a general rule it is better to start with higher values than with lower ones since the scheme is biased towards eventually lowering values in the later part of adaptation. However, a better solution might be to institute a new scheme during the first 10 or 20 generations whose sole purpose would be to extract a set of parameter values to be used from then on. During these initial generations, the genetic operators could be applied at random or in some systematic way so as to test as many parameter value combinations as possible.

Another problem involves the crossover operators. As we explained above, the parameter values for these operators fluctuated more than desired, especially in the later stages of adaptation. A relatively simple way to reduce this phenomenon is to readjust the PGAP factor (see presentation of Schemes 1 and 2 above). By imposing a maximum of .6 or .8 on each crossover operator we would reduce the size of PGAP *when an increase in probabilities is due*. Furthermore, this maximum would guard against extreme crossover rates which, as we mentioned above, can come about due

to loss of variance in the population. We could also reduce the maximum on PGAP from .5 to .25. Again this would affect increases more than decreases since PGAP is usually less than .25 when a decrease in probabilities is in order.

The next step in improving the modification of crossover probabilities would be to maintain only one probability to determine the application rate of a general crossover operator. The question of how many breaks will occur could be decided independently by examining only those individuals which were produced using crossover. In this way the crossover probability would be modified on the basis of twice as many samples as before since previously only one of the two crossover operators could be applied. This would definitely reduce some of the fluctuations.

A way to check for unjustified rewards to the crossover operators and also help to minimize problems involved with loss of variance, would be to maintain a similarity measure between individuals. If an individual were produced using only mutation, it would be very similar to its parent. In this case crossover could be prohibited between these two individuals (a sort of incest taboo) or the crossover could carry little weight in modifying the crossover probability. Another possibility would be to eliminate the inferior member of a very similar pair, thereby maintaining a rich variety of individuals. This method will be investigated in Section 8.4.

Another problem with our parameter modification scheme involves interactions between operators. For example, if a new individual is formed using crossover and some mutation operators, it is very likely that its performance will be due mainly to crossover, which made the largest modification. Therefore, the mutation parameters should be modified very

little based upon this sample. There are probably similar interactions between the mutation operators but to a much lesser degree.

Our final comment involves the false peak phenomenon in the meta-space of parameter values. About the only time this becomes a problem is during the later stages of adaptation. Take the following observed situation. Low crossover and mutation rates produce many similar individuals. Inversion is also low since it equals the average of the two crossover operators. Soon crossover takes place between identical segments on two different strings. No actual change takes place but the offspring performs as well as its parents and the crossover operator is rewarded. Thus the crossover rate increases pulling inversion up with it. Increased inversion makes it less likely that crossover will occur between identical segments. But now almost every offspring is formed via crossover and the modification is too large (at the later stages of adaptation) for it to perform well enough to replace a population member. Without new population members no change occurs in the high crossover values and almost all new individuals are inferior.

This problem might be eliminated using some of the suggestions above (e.g., maximum crossover rate, similarity measures). However, there are other solutions. One would be to use information from all new individuals to modify parameter values especially when there are few new population members. Another solution would be to introduce a low level of random applications of operators throughout adaptation to avoid extreme dependence on the current parameter values. Either of these solutions would have most likely avoided the above problem.

Summary

In this section we have successfully developed a parameter modification scheme which resulted in more efficient and effective use of reproductive plans.

The scheme maintained one set of parameters for all individuals and used observed operator frequencies in new population members to modify the current parameter values. In addition, some biases were introduced to lower parameter values near the end of adaptation.

The scheme was evaluated and developed with respect to stability and convergence criteria. We found that it was capable of converging on final mutation values which were significantly different for the different mutation operators and for different chromosome lengths but which were not affected by population size or starting parameter values. Similar trends did not occur with the crossover operators for reasons which we have discussed above.

Plans using the parameter modification scheme outperformed similar plans without the modification schemes. This result was particularly evident in the long run and was upheld over different chromosome lengths, starting parameter values, and plan modifications.

Finally, we have discussed some of the bad aspects of the parameter modification scheme and suggested possible improvements.

The successful implementation of a parameter modification scheme is a major step in developing efficient and effective reproductive plans. First, it eliminates all the testing that took place in the previous section to find "optimal" parameter settings. This is important since these plans typically require a large amount of time. Secondly, it enables the plan to use what operators it needs when it needs them. This is very impor-

tant since the need for different operators at different levels of evolution has been demonstrated previously in this task and in earlier work. Finally, by examining the changes in operator parameters, the researcher can gain valuable information about the space he is searching even though he still receives payoff-only information from the environment. This could eventually result in the development of valuable new operators.

Table 7.2.8 indicates how the formal elements of an adaptive plan, $M_t, \tilde{\tau}$, and m have been modified or extended as a result of the parameter modification scheme. Note that most of the work was involved in developing a good function m to change the current set of parameters in memory.

Formal Element	Modification Involved
M_t: Current Memory	1) Current set of parameters 2) Current observed frequency of operator usage in new population members 3) Number of new population members since last change in parameter settings
$\tilde{\tau}$: Device Selection function	Uses *current* parameters for recombination
m: Memory updating function	1) Update current parameters (various schemes were tried to reduce fluctuations and produce consistent performance) 2) Keep track of operator usage and new population members

Table 7.2.8 Changes Due to the Parameter Modification Scheme

7.3 Some Additional Observations on Population Size and Crossover

Before ending this chapter we would like to reassess the effects of population size and the crossover and inversion operators in light of the experiments of the previous section and the introduction of the parameter modification scheme.

Table 7.3.1 presents three experiments from the previous section which differ only in the composition of the population used. All experiments used the parameter modification scheme on the difficult task. We have included the number of samples before shutoff to adjust for the different sampling rate of the 20/10 population. The 12/6 population proved to be significantly inferior to both of the other populations. The comparison with the 20/10 population is not justified due to a significantly different number of samples. However, the 20/14 populations did not use significantly more samples but did produce significantly better results. Comparisons between the 20/10 and 20/14 populations indicate no significant difference in performance even though the 20/10 population used significantly more samples. These results point towards to 20/14 population for future use. It performs at least as well as the 20/10 population yet it has a larger population base (important for maintaining population variance.)

EX NO	POP SIZE	STOP MEAN	STOP GEN	NUMBER SAMPLES	SIGN
11	12/6	61.4	99	594	*
13	20/10	70.2	102	1020	
15	20/14	67.5	120	720	

Table 7.3.1 Population Size Comparison

Now let us reexamine the effects of crossover and inversion. Experiments in Section 7.1 seemed to indicate that these operators were a valuable addition to reproductive plans. However, the results of the last section did not verify the plan's need for definite crossover application levels at certain points in evolution. Therefore, we have run some additional experiments using the parameter modification scheme but without the use of the crossover operators or inversion. In these experiments the parameter modification scheme should work even better on the mutation rates since there is no crossover to mask the effects of good or bad mutations. Starting mutation rates were the same for all experiments.

Table 7.3.2 summarizes the results of two pairs of experiments. The entry under C&I? indicates the use of crossover and inversion. Experiments 17 and 27 were run with a short chromosome (17 detectors). Experiment 17, using crossover and inversion, was significantly superior (at better than the 1% level) for every comparison point used. Experiments 25 and 26 involved one of the best reproductive plans developed (in next chapter) run with and without crossover. Again the plan with crossover was superior at the same significance levels. These results provide solid evidence that crossover and inversion are extremely valuable mixing operators independent of chromosome size and plan modifications.

This section ends our study of the basic elements of reproductive plans. The rest of our work will be devoted to exploring a number of schemes to further improve the operation of these plans. We shall be principally concerned with maintaining an effective search for an extended period of time. Therefore, many of the schemes will be used only

EX NO	C&I?	STOP MEAN	STOP GEN	GEN 150	GEN 200	SIGN
17	Yes	71.0	122	72.2	73.2	*
27	No	62.0	115	63.5	64.8	
25	Yes	75.5	130	76.8	79.2	*
26	No	64.5	103	68.2	71.1	

Table 7.3.2 The Effects of Crossover and Inversion

when adaptation slows down or when the population lands on a false peak. As a means to the above end we will also be concerned with maintaining a rich *variety* of individuals in the population since loss of variance is often the major cause of false peaks. Let us now turn to this endeavor.

Chapter 8 Further Refinements of the Reproductive Paradigm

Introduction

In this chapter we will investigate a number of techniques to further improve the performance of reproductive plans. Basically, we will be concerned with inducing an effective search. In other words, we will try to develop a plan which is capable of maintaining significant adaptation over an extended period rather than operating efficiently for a short period. Therefore, many of the schemes which we will try will take effect only when performance seems to be leveling off.

Figure 8.1.1 presents a flow diagram outline of the current status of the reproductive plan. In the large box we have indicated how the parameter modification scheme affects the process of generating new offspring. In the top left hand portion we have added a new option called the variable chromosome scheme. Let us now turn to this scheme.

8.1 Variable Length Chromosomes

The purpose of this scheme is to force an individual to use all of its detectors effectively by initially giving it only a few detectors to use. The plan proceeds as follows. At the beginning of adaptation, all chromosomes contain about one-third of the normal number of detectors. Adaptation continues with these chromosomes until the marginal gain in the average performance of the population during the previous 15 generations falls below a certain level called the *extension level*.

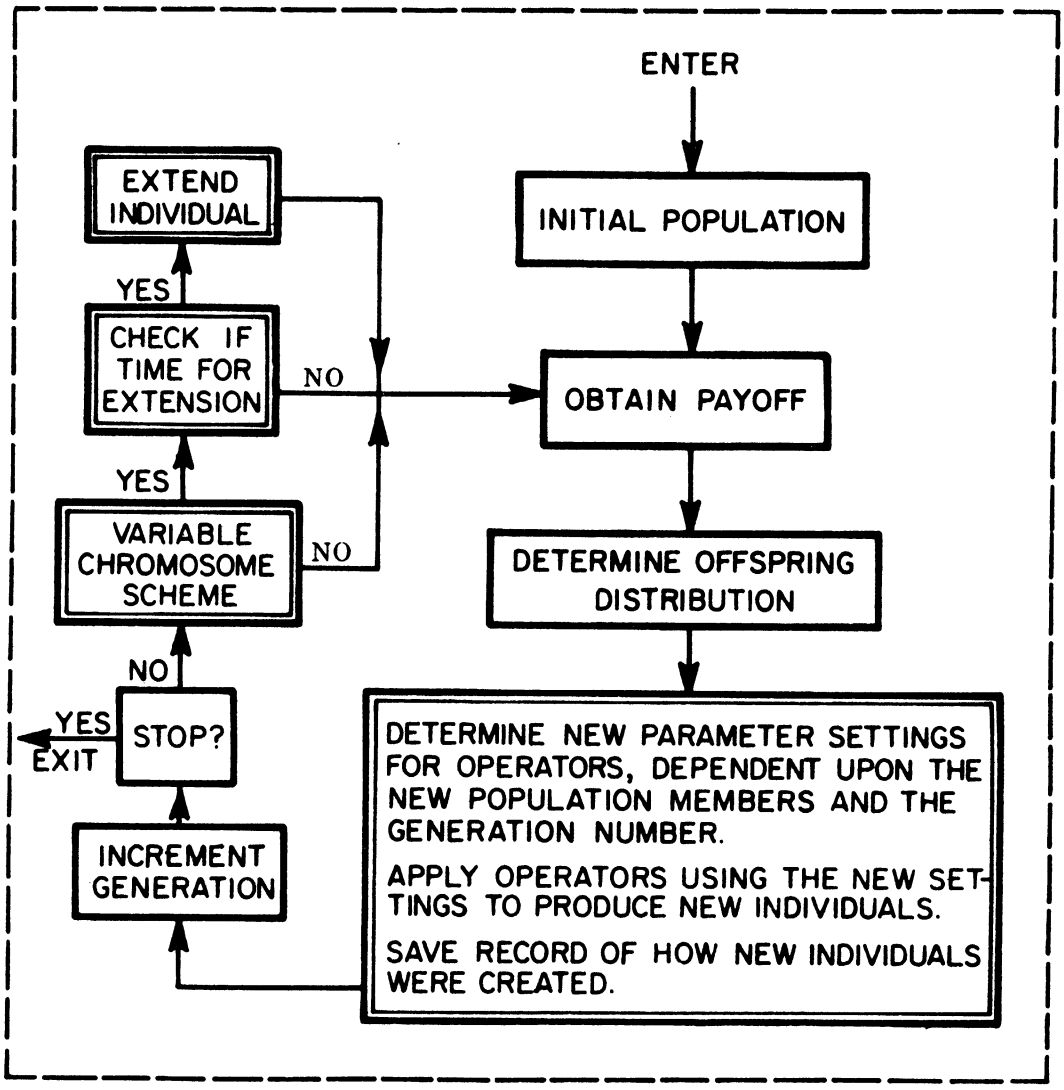


Figure 8.1.1 Current status of the investigation

At this point more detectors are added to each individual and the whole chromosome is retested to receive a new utility. Adaptation with this "new" population continues for at least an additional 15 generations and then as long as the marginal gain remains greater than the extension level. This process is repeated adding more detectors until all chromosomes have the normal amount. Then normal adaptation is continued until some stopping rule is exercised.

In developing this plan we have assumed that an individual which is forced to use limited resources will build up valuable small subsets of detectors. When these resources seem to be insufficient for further improvement we provide for additional resources which presumably will induce a sudden increase in performance. By the time the individual is using the normal number of detectors he will hopefully have built up a more powerful set than would be the case if he were using all the detectors from the start.

In developing this scheme we will be concerned with how the extension will be made (i.e., where we will obtain the additional detectors) and when the extension will be made (i.e., what extension level). A chromosome normally contains 110 detectors. Each chromosome will consist of 35 detectors initially, and will gain an additional 25 during each extension for 3 extensions.

Scheme 1: Random Extension

In this scheme the extension level was set at 4.0. When the change in utility for the previous 15 generations fell below that level the chromosomes were extended by adding randomly generated new

detectors.

The results of this scheme were not favorable. Every time a chromosome was extended it experienced an immediate *decrease* in utility contrary to what we had expected. Obviously, the inferior performance of randomly generated detectors outweighed the benefit of having more detectors. It took the plan on the average 15 generations to regain previous utility levels after the addition of the new detectors. Using the slope stopping criterion this scheme turned in performance comparable to the constant chromosome scheme (i.e., using 110 detectors from the start). However, the random extension scheme used about 50 additional generations before stopping (significant at the 1% level). This suggests that the random extension scheme is not a valuable addition to reproductive plans.

Scheme 2: Intrachromosomal Duplication

A process called intrachromosomal duplication has been observed in the real world. As a result of this process, a gene or set of genes becomes duplicated many times within a chromosome. This effectively increases the local sampling rate of slight variations of this gene since it is more probable now that one of the duplicates will undergo mutation.

This operation seemed ideal for our situation. The random extension scheme probably failed due to the jarring effect of adding random detectors to detectors which had worked very well together. Intra-chromosomal duplication would add additional genes without having this jarring effect since the "new" genes would only be copies of others.

This is analogous to attaching weights of 2 to some of the detectors in the pattern recognition task. This probably will affect the new utility a bit depending on whether the duplicated detectors were good or bad, but generally performance should not change greatly and when it does, it sould increase as often as it decreases.

Results with this scheme were about as expected. Although performance after extension was often below that before extension, it only took 4 generations on the average to regain the previous level. Furthermore, final performance under the slope stopping criterion was superior to that of the constant chromosome scheme. These results will be presented later in table form.

Since this scheme was so successful we decided to test the effect of different extension levels. Low extension levels (2.0, 3.0) seemed to force the individual to become too good before extension. As a result, performance dropped by a greater amount after extension and it took more generations to regain previous levels. The net result was a larger number of generations before stopping, but no similar increase in performance. An extension level of 8.0 seemed to be the best. Performance was similar to experiments using an extension level of 4.0 but fewer generations were needed.

Scheme 3: Concatenated Segments

This scheme differs from the above in that the chromosome segments initially evolve independently of one another. At the beginning each of three independent populations of chromosomes (containing 36 detectors) evolve until marginal performance falls below the extension level. At this point a population of normal length chromosomes is generated by

randomly selecting one segment from each of the three populations and concatenating the detector strings. This new population then becomes the current population. However, for at least the next three generations, new individuals are still formed using concatenated segments, not by the usual duplication and recombination methods of reproductive plans. After these three generations, this process is continued as long as at least one *new population member* is created per subsequent generation. As soon as this condition is not met, the three populations of concatenated segments are discarded and the normal operation of the reproductive plan takes over for the rest of adaptation. Care is taken that no concatenated individual is identical to a previously created concatenated individual.

It was felt that increasing the sampling of concatenated individuals beyond the formation of the new population was a worthwhile expenditure since it constantly increased the mean performance of the population without any loss in variance. In this respect, each individual differs from another in at least one whole segment, i.e., 1/3 of the chromosome. The increased sampling was justified. As opposed to the previous schemes where extension typically resulted in temporary falling off of performance, the concatenation scheme resulted in a population whose mean performance was significantly higher than the mean performance of all segments contributing to the concatenation process. Furthermore, the best individual of the concatenated population was always superior to the best of all segments contributing to the population. Since we have shown using previous schemes that size alone does not account for the goodness of a chromosome, we must

conclude that the concatenated segments work in somewhat of a complementary fashion. Results also show that on the average seven additional generations were spent sampling concatenated individuals after the initial three. Again we found that a high extension level (concatenating early in evolution) produced better end results in fewer generations.

Quantitative Analysis of Variable Chromosome Schemes

Let us now compare the performance of all three schemes to that of the constant chromosome scheme. Table 8.1.1 gives performance based on the slope stopping criterion. Both Schemes 2 and 3 significantly dominated the constant chromosome scheme. The significance was less for the random extension scheme due to its large variance. In fact, the variance of Scheme 2 (intrachromosomal duplication) was significantly lower than that of Scheme 1. Furthermore, Scheme 1 took significantly more generations than did the constant chromosome scheme or Scheme 2. The number of generations shown for Scheme 3 is a bit deceptive since we have included all generations of all three initial populations of segments.

EX NO	EXTENSION SCHEME	STOP MEAN	STOP GEN	STOP S.D.	SIGN
1	Const.Chrom.	67.5	120	4.5	
2	1-Random	65.3	167	8.2	
3	2-Inchrom. Dup.	71.5	126	3.4	*
4	3-Concat. Seg.	71.7	149	3.6	*

Table 8.1.1 Summary of Variable Chromosome Schemes

The above results indicate that two of the variable chromosome schemes, intrachromosomal duplication and concatenated segments, perform significantly better than the best constant chromosome scheme. We have chosen to maintain the intrachromosomal duplication scheme for further experiments. Although the concatenated segments scheme performed as well as the intrachromosomal duplication scheme, the former required more controls and used more generations.

Effects of Population Size: Revisited

Before ending this section we would like to present the results of some experiments which again test the effect of population size. Since the beginning of our investigations, we have found that increasing population size has had a favorable effect on performance even though we maintained the same number of samples per generation. Moreover, the introduction of the parameter modification scheme and the intrachromosomal duplication scheme has increased the need for a large population in order to maintain sufficient population variance. The parameter modification scheme reduces population variance as a result of low parameter settings during the later stages of adaptation. In this situation many offspring closely resemble their parents and replace some other individual in the current population, thereby increasing the homogeneity of the population. Intrachromosomal duplication can produce a similar effect due to the duplication of genetic material within an individual. The combination of these two schemes can severely inhibit significant adaptation during advanced stages due to the loss of variance and the resultant loss of a parallel search.

We have introduced the 40/34 population to further guard against loss of variance. In addition, we have modified our selection scheme when using this population. If we allowed all 34 members to produce modified utilities for use in offspring distribution, all individuals would be apportioned a small fraction of an offspring. As a result, all six offspring would be randomly distributed over the population resulting in an underselection situation. Therefore, we shall use only the top 14 individuals for the initial weight assignment of offspring. Any excess offspring due to an individual exceeding his quota during one generation or over all generations will be distributed randomly among all members of the population. If there were no excess offspring this population would act exactly like the 20/14 population since the extra population members would never by able to produce offspring and, therefore, might as well not be there.

However, table 8.1.2 indicates that the 40/34 population does have a significant effect. For the constant chromosome scheme, there was no clear cut effect. Experiment 5, using a 40/34 population, performed very erratically with many runs terminating very early. As a

EX NO	POP SIZE	SCHEME	STOP MEAN	STOP GEN	STOP S.D.	SIGN
1	20/14	Const. Chrom.	67.5	120	4.5	
5	40/34	Const. Chrom.	60.4	101	11.2	
3	20/14	Ichrom. Dup.	71.5	126	3.4	
6	40/34	Ichrom. Dup.	75.5	130	4.3	*

Table 8.1.2 Effects of a Large Population

result, the variance of this experiment was very large and significantly different from all others. There was no significant difference in mean performance between experiments 1 and 5.

The intrachromosomal duplication scheme, on the other hand, reacted very favorably to the larger population. As evidenced by experiment 6, it produced performance which was significantly superior to all other experiments (at the 2% level) without any significant increase in the number of generations. Experiment 6 has proved to be one of the most successful reproductive plans that we have developed. It has been referred to in earlier chapters when we have performed final tests on various schemes. In these cases we have run experiment 6 with one modification (e.g., a different selection scheme or without the parameter modification scheme). In all cases the modification has resulted in performance inferior to experiment 6. These results indicate that the reproductive plan used in experiment 6 consists of many parts, all of which are needed to turn in the excellent performance that we have seen.

A few additional experiments were run to test the effect of population size and sampling rates in extreme situations. These experiments used 4/2 and 40/38 populations. Runs were maintained for 600 generations so as to permit comparison with other experiments having a higher sampling rate.

Table 8.1.3 summarizes the results. Besides maintaining a constant chromosome, experiment 7 did *not* use the parameter modification scheme; the other experiments did. For all comparisons, there was no significant difference between experiments 7 and 8. This may seem surprising

EX NO	POP SIZE	SCHEME	GEN 200	GEN 450	GEN 600	SIGN
7	4/2	Const. Chrom.	53.8	61.1	63.5	
8	4/2	Ichrom. Dup.	52.3	58.2	60.1	
9	40/38	Ichrom. Dup	64.3	75.3	77.3	*

Table 8.1.3 Effects of Different Population Sizes

since we have already shown that the parameter modification scheme and the intrachromosomal duplication scheme produce superior results. However, these experiments point out the importance of a large population in these cases. Experiment 9 supplies the most convincing evidence since it differs from experiment 8 only in the size of the current population. This was significantly superior to the other experiments at the 2% level. As far as sampling rates are concerned, there was no significant difference between experiment 9 (2 samples/generation) and experiment 6 (Table 8.1.2, 6 samples/generation); comparisons were made using equal numbers of total samples.

Summary

In this section we have investigated a different type of reproductive plan. The device selection function $\tilde{\tau}$ did not change. However, the search space $\mathcal{A}$ was modified (expanded) as the search proceeded. Some heuristic techniques have already been developed which increase the

efficiency of a search by effectively decreasing the size of the search space [Lin]. The variable chromosome scheme has achieved a similar goal but from a different direction. Initially, the search space is small. Once some peaks in this space have been discovered, the space is expanded so that the previous peaks are preserved in the larger space.

It was discovered that intrachromosomal duplication and concatenating segments were both successful in preserving previous peaks in the expanded space. Furthermore, these schemes performed significantly better than the constant chromosome scheme without any increase in the number of samples. We will use both the constant chromosome scheme and the intrachromosomal duplication scheme for future experiments.

In addition to testing variable chromosome schemes, we have also extended our tests on the effects of population size and sampling rates. In these experiments we found that a 40/34 population can further improve the performance of schemes which normally result in a decrease in population variance. On the other hand, our most powerful schemes were shown to be of no benefit when 4/2 populations were run. This result is very significant considering that *Fogel et al*, Klopf, Samuel and Uhr and Vossler all maintained "populations" of only one or two individuals.

8.2 Auxiliary Populations

In this section we will experiment with two techniques which maintain or generate more than one population of devices in the course of evolution. Again the main purpose of these techniques is to maintain significant adaptation over an extended period of time.

Isolated Populations

Our first technique is to operate two separate populations until performance appears to be leveling off. At this point the populations are mixed to form two new populations, each new population containing some individuals from each of the isolated populations. Then the new populations are permitted to continue adaptation, hopefully more successfully than the previous populations.

In using this technique, we are assuming that performance usually levels off because a population loses much of its variance and thereby loses its ability to carry on a truly parallel search. Since many individuals closely resemble one another at this point, crossover fails to bring about any meaningful mixing, leaving mutation as the only means of variation. However, by mixing populations which have evolved independently of one another, we will end up with genetically different individuals which have comparable payoffs. In this case, it seems very likely that crossover will produce valuable exchanges and the parallelism of the search will be maintained.

The scheme was implemented in the following manner. The first population evolved until its marginal utility fell below a certain level. At that point, the population was saved along with its mean performance. The second population evolved until its mean performance was just greater than the mean performance of the first population. At this point, the two populations were combined to produce two mixed populations. The first, third, fifth, ... individuals of the first population were combined with the second, fourth, sixth, ... individuals of the second population to form one mixed population, the other mixed population being formed similarly. Then each mixed population evolved independently for at least fifteen generations and then until the marginal utility stopping rule was exercised.

The above scheme was used to assure that both isolated populations reached comparable performance levels before mixing. Otherwise individuals from a superior population would dominate those from an inferior population and no significant mating would take place in the mixed population. Populations of size 20/14 were run using the parameter modification scheme with constant length chromosomes.

For the first few runs, the first population evolved until its marginal utility for the previous 15 generations was less than 2.0. This, however, resulted in population mixing at too early a point in evolution. Significant adaptation did continue after mixing but it was hard to say whether this was due to the mixing or to the fact that there still was plenty of room for improvement. At any rate this did not address itself to the real problem, namely that of forcing all runs

to attain nearly the same high degree of performance.

Therefore, the first population was allowed to evolve until its marginal utility was less than 1.0. Since this is the normal stopping point for our runs, any additional increase above the performance level attained by these isolated populations should be considered important.

The results, however, were disappointing. Of two runs tried, only one of the four mixed populations continued to any substantial degree past the 15 generation minimum after mixing. In fact, this one population remained mixed for only three generations after the initial mixing, whereupon all individuals from one of the isolated populations were selected out in favor of the individuals from the other population and their offspring. In addition, we discovered that little mixing took place via the crossover operators. For two of the mixed populations, no new population members were formed from a crossover between members of different isolated populations. Another mixed population generated one "mixed individual" just before the run was terminated. The fourth population produced a few "mixed individuals" but still could not increase its mean performance enough to avoid being shut off after 15 generations. In light of this evidence, no further runs were made using this particular scheme.

The cost of running two isolated populations is certainly great; therefore, one should be guaranteed success almost every time. One way to reduce the cost factor would be to run the two isolated populations sampling only half as many individuals as usual. This would solve the cost problem but two difficulties remain. The performance of the isolated populations with half the sampling would certainly be inferior

to a population sampling twice as much. Secondly, the problems after mixing still remain.

Unless significant mixing takes place between individuals from different populations through a crossover-like operator, isolated populations will serve no purpose. One of the problems with our scheme might have been due to the lack of rigid controls after the mixed populations were formed. At this point, it might have been helpful to aid the process of forming a truly mixed population (of mixed individuals) by requiring all offspring to be formed via crossover. If this proved to be too great a change, then biases could be placed on the breakage points for double crossover so that the length of the exchanged segments was relatively small. After a number of fairly good mixed individuals were formed, these controls could be relaxed and adaptation could continue normally.

Another technique which might help the isolated populations scheme would be to mix the populations early in evolution, force the formation of mixed individuals, and then continue to mix the newly formed mixed populations at selected points thereafter. This technique would probably result in more similarity between the isolated populations but would still maintain added parallelism in the search. Furthermore, the similarity between the isolated populations would make it increasingly likely that good mixed individuals would be formed each time the populations were mixed.

New Populations

Instead of implementing the above modification of the isolated populations scheme, we decided to try a new scheme which hopefully will serve the same purpose, i.e., to force all runs to discover significant peaks in the search space. This scheme operates in the following manner. At each generation a copy of an individual randomly chosen from the current population is stored in an auxiliary population. This secondary population does not come under selection but is continually modified by replacing the "oldest" individual every generation. When the performance of the current population seems to be leveling off, the secondary population becomes the current population and takes over the evolutionary process. The other population is discarded. This new population continues adaptation for at least as many generations as is needed to build up another secondary population. Then it continues on its own merit until its average performance falls below a specified level.

For the first few runs populations were replaced when the marginal utility was less than 1.0. However, few new populations were able to survive at this pace and were replaced as soon as another new population became available. This was not desirable. Therefore, we ran the new populations until the marginal utility fell below .5 and stayed below .5 for up to four successive generations. This latter condition was useful since it allowed *momentary* dips below the crucial value.

With this new replacement rule, many of the new populations did remain in use for a substantial period of time on their own merit.

The problem now was to determine an adequate stopping rule for each run. We decided to permit a total of three new populations. Since each auxiliary population is generated from the previous population and its offspring, there is usually a loss in variation with each new population producing a decreasing marginal utility. In addition, we felt that if any gains were to be made they should have taken place by the third new population.

This scheme was tested with a 20/14 population using the parameter modification scheme. Both constant chromosomes and varying chromosomes (using intrachromosomal duplication) were used. The results of these runs were for the most part difficult to analyze. To reiterate, the purpose of this scheme is to enable evolution to continue when it appears that the present population has reached a peak. Therefore, we should at least expect a significant difference in performance levels after the third new population when compared to the level before the first new population. However, some compensation must be made for the cost of running these additional populations. In particular, each new population will run a minimum of 14 generations. In a total of 42 additional generations it is probable that the original population would have made some additional gains. Therefore, comparisons must take into account the total number of generations elapsed before final payoff.

With this criterion in mind we shall now evaluate the new-populations technique. The experiments with the intrachromosomal duplication scheme seemed to indicate that the new populations added very little. Of six runs, four realized very little or no increase in performance

due to the new populations. However, there was an unexpected complication in these tests since two of the runs attained very high levels of performance before the new populations were used. Therefore, there was little room for additional improvement. Another two runs attained comparatively low levels of performance but still were not able to benefit from the new populations. However, upon investigation of these runs we found that the original populations had lost most of their variance well before the new populations took over. As a result each successive population was more and more homogeneous. Two other runs did achieve a relatively larger performance increase with the new populations, although the final performance levels were not above what is typically reached using the same total number of generations. Therefore, one can conclude that the new population scheme is not worthwhile when used with the intrachromosomal duplication scheme.

Using constant chromosomes, the results were a bit easier to analyze. All the runs attained comparable performance levels before the new populations and all realized some gain with the new populations. In particular, the best run before the new populations realized the least improvement by means of the new populations, while one of the worst runs before the new populations realized the most improvement. This certainly seems to satisfy the desire of bringing inferior runs up to the level of other runs.

Table 8.2 gives average performance levels right before replacement of the original population and after each new population. The gain in performance with each new population is not overly impressive, but does indicate that the new populations are capable of sustaining

STATE OF THE SCHEME	AVE GEN	AVE PAYOFF
Before first new population	183	72.8
After first new population	212	73.8
After second new population	240	74.7
After third new population	287	77.4

Table 8.2 The effect of new populations on performance using constant length chromosomes

adaptation. However, the current method of generating new populations is rather crude.

A refinement of the new population scheme might prove more successful. For one thing at least one of the new populations in each run achieved little or no improvement whatsoever. Yet, because they were required to run for a minimum of 14 generations they did succeed in diluting the performance of the better populations since total time is taken into account. Therefore, any improvement in the selection of individuals for the new populations might prove helpful. One such improvement would result from checking for duplications during the process of randomly picking new population members. Some duplications are highly probable and serve only to reduce the population variance.

Summary

In this section we have investigated two schemes in an attempt to

maintain significant adaptation over an extended period of time. However, we have come to the general conclusion that the schemes did not live up to our initial expectations.

We have presented suggestions at the end of each scheme as to how we feel the schemes could be improved. In general, these suggestions call for more elaborate controls. We have developed reproductive plans which are now working very well. Therefore, it is difficult to get them to perform even better without sophisticated controls. Such controls might involve monitoring the population to assure that variation is maintained.

8.3 Mutation Pools

In the previous section we tried to maintain significant adaptation by operating isolated populations or generating new populations. However, the isolated populations did not mix sufficiently and the new populations were probably formed too late in evolution after most of the damage (i.e., loss of variance) had been done.

In this and the next section, we will investigate techniques which will operate continuously as the population evolves. First we will try a scheme, similar to dominance in the real world, which should maintain rapid access to potentially valuable genes or detectors. Then in the next section we will continue our efforts to maintain a rich mixture of different individuals in the population.

Let us briefly describe the role of dominance in a real world

situation. In the typical case we are dealing with a diploid individual, which possesses a number of *pairs* of chromosomes. Therefore, each gene is composed of two possibly different alleles. In the case of pure dominance, only one of two different alleles has an effect on the individual's performance. This one is called the *dominant* allele while the other is considered *recessive*.

One benefit of this situation is that it reduces selective forces against a number of recessive alleles. If each allele were expressed according to its worth in the present environment, selection would often favor the "best" one, eliminating the rest. Two possible dangers may result from this situation. First, the environment may change so that an allele which was very good in the previous environment may only be average or even lethal in the new environment. Secondly, there might be interactions between genes so that a particular allele might be best only while existing simultaneously with another allele at a different gene. In other words, the goodness of an allele may be very dependent upon the context in which it appears. If this context changes due to the change in an allele at another gene, the particular allele which formerly appeared to be very good may not perform very well any longer.

Without dominance in an isolated population, the only way to replace lost alleles is through mutation. However, mutation is for the most part a random operation which provides for very slow replacement of alleles. In a dominance situation the recessive allele is carried along with the dominant one but is not subject to selective forces

since it does not affect performance. However, if identical recessives show up on both genes of a chromosome pair they will affect performance and be subject to selection. In this way dominance provides for rapid access to a number of different alleles.

Dominance itself may also change due to changes in the environment. In other words, an allele which in one environment played a dominant role may eventually become recessive in a different environment. The phenomenon of industrial melanism provides the most convincing example of how a change in environment can force a change in dominance (Wallace, 1968).

It certainly seems that dominance would be a valuable addition to reproductive plans. However, with our task there is no important distinction between a gene and an allele and it is doubtful that any single detector will play a large enough role so that over-selection will take place over the number of generations we are dealing with. Yet, the ability of the dominance situation to provide rapid access to alleles in addition to mutation seems to be an interesting technique which we could use.

Basically we will try to improve the performance of the Mutation 2 operator (which previously produced randomly generated detectors) by maintaining a recessive pool of detectors which are extracted from the current population throughout evolution. The detectors in this pool are not affected by selection, but when the plan calls for the use of mutation 2, a detector randomly taken from the pool will replace a detector on a chromosome. Thus the previously "recessive" detector is now affected by selective forces.

Let us look at the specific implementation of the mutation pool. The pool actually consists of five subpools, each subpool containing

about 100 detectors. After the initial population is randomly generated and tested, the pools are filled by randomly extracting detectors from the top portion of the population. Most likely there will be duplicates. Thereafter each subpool is used once every five generations and serves as the source of genetic material for Mutation 2. During each generation one-fifth of the detectors in the current subpool is replaced with detectors extracted from the current population. Detectors in the subpools are dated so as to replace the oldest ones first. Therefore, at each generation detectors are available from a subset of the past 25 generations.

Our first experiment used mutation 2 alone to test the effect of the pool without the interference of other operators. For comparison with a previous experiment we used a 12/6 population on the easy task. Experiments 10 and 11 in Table 8.3 show the results of this test. Mutation 2 operating from the pool produced significantly better adaptation than random Mutation 2. Next we ran our 20/14 population on the difficult task using all genetic operators. Experiment 12 did not use

EX NO	POP SIZE	TASK	PARAM MOD?	MUT 2 POOL?	STOP MEAN	STOP GEN	SIGN
10	12/6	EASY	NO	NO	71.1	43	
11	12/6	EASY	NO	YES	77.3	55	*
12	20/14	DIFF.	YES	NO	67.5	120	
13	20/14	DIFF.	YES	YES	68.2	126	

Table 8.3 The effects of the mutation pool

the mutation pool while 13 did. The lack of significance between these two experiments is probably due to the fact that Mutation 2 plays a relatively minor role in the evolutionary process when compared to the role of the other genetic operators. However, we have shown that the recessive pool does improve the performance of Mutation 2.

The mutation pool can be viewed as a method which eliminates some of the random aspects of mutation by directing the choice of new alleles towards those which are more likely to produce high payoff. Mutation 1 could be modified similarly. Instead of replacing an n-tuple point with a randomly chosen mesh point, we could extract some random point from another detector in the current population. Alternatively, we could bias our choice of a new mesh point by picking new mesh points which are spatially close to the point we have eliminated.

8.4 Maintaining Population Variance

Introduction

In the previous section we were concerned with maintaining variance at the gene level. In this section, we will be concerned with investigating a technique for maintaining population variance; i.e., maintaining a rich variety of individuals in the population. Rather than trying to correct for the loss of population variance as we did in Section 8.2 on auxiliary populations, we will apply a more continuous

effort to insure that no two individuals in the population are too similar to one another.

Near the end of Chapter 7 we commented on the need for a method which would maintain population variance. At that time we suggested calculating a similarity measure between individuals in the current population and using this measure to influence pairing for the cross-over operation or to determine which population members should be discarded. However, it would be very costly to calculate a similarity measure for each new population member. First the measuring process itself could prove lengthy. For example, a suitable measure should take into account the number of identical detectors existing between two individuals. However, we could not calculate this just by lining up the strings and checking for matching detectors along the length since an inversion might have reversed positions. Therefore, we would have to use a more sophisticated process and, for each new individual, we would have to apply this process to all N-1 pairs of individuals in a population of size N.

In order to avoid the very high costs associated with the above process, we will use less exact but simpler techniques. We assume that a similarity measure will be highest between parents and offspring. Therefore, we will check for similarity only between these individuals. In this way, our similarity measure can be based upon which genetic operators were used to form the offspring rather than upon structural similarities between all individuals. There is also a good chance that different offspring from the same parent will be similar. However,

this chance will be minimized as we will demonstrate below.

Before presenting the three schemes that we have investigated, we would like to comment on how we should evaluate them. These schemes constitute a major change in the operation of reproductive plans. What we will do is add a preselection scheme so that individuals will be discarded using information in addition to payoff. However, many of the techniques we have previously incorporated (e.g., parameter modification) were developed using only our previous selection scheme. Therefore, there is a good chance that performance using our preselection schemes may appear inferior even though it could prove to be superior in the long run.

In this regard we will be satisfied if performance with our new schemes is only as good as performance with our old scheme since as a valuable addition we will have evolved populations with a larger variety of individuals. We have previously mentioned (Section 4.2) why population variance is important for successful operation of reproductive plans. This variation is important for a number of additional reasons. First, the greater the variation the more likely that the population will continue to improve past the final point that we use for our comparisons. Secondly, we will have a number of truly different devices which are very capable of performing a given task. This would be valuable, for example, if the environment changed. In this case there is a greater chance that some devices will "survive" the change in environment and continue to receive a high payoff in the new environment. If many of the devices are similar before the change in environment, they will probably perform similarly (good or bad)

after the change. Finally it might just prove convenient for some researchers to have a number of different good devices rather than just a few.

Preselection Schemes

We will call our schemes preselection schemes since they may dispose of individuals (either offspring or parents) before the normal selection scheme is used. Our previous selection scheme will not change. Figure 8.4 shows where this preselection process fits into the operation of our reproductive plans. Below we will present three schemes. Each scheme will be run with a 20/14 population using constant sized chromosomes. Then we will compare performance using these schemes to the performance of constant chromosomes without any of these schemes and to the performance of the interchromsomal duplication scheme, which to date has produced the best reproductive plan.

Scheme 1

This scheme exerts the least selection of the three we will present. Its development was prompted by the following observation. It often was the case that an offspring performed worse than its parents (or parent) but, nevertheless, performed well enough to replace a current population member. In other words, the mutation and/or crossover was harmful but not harmful enough for the offspring

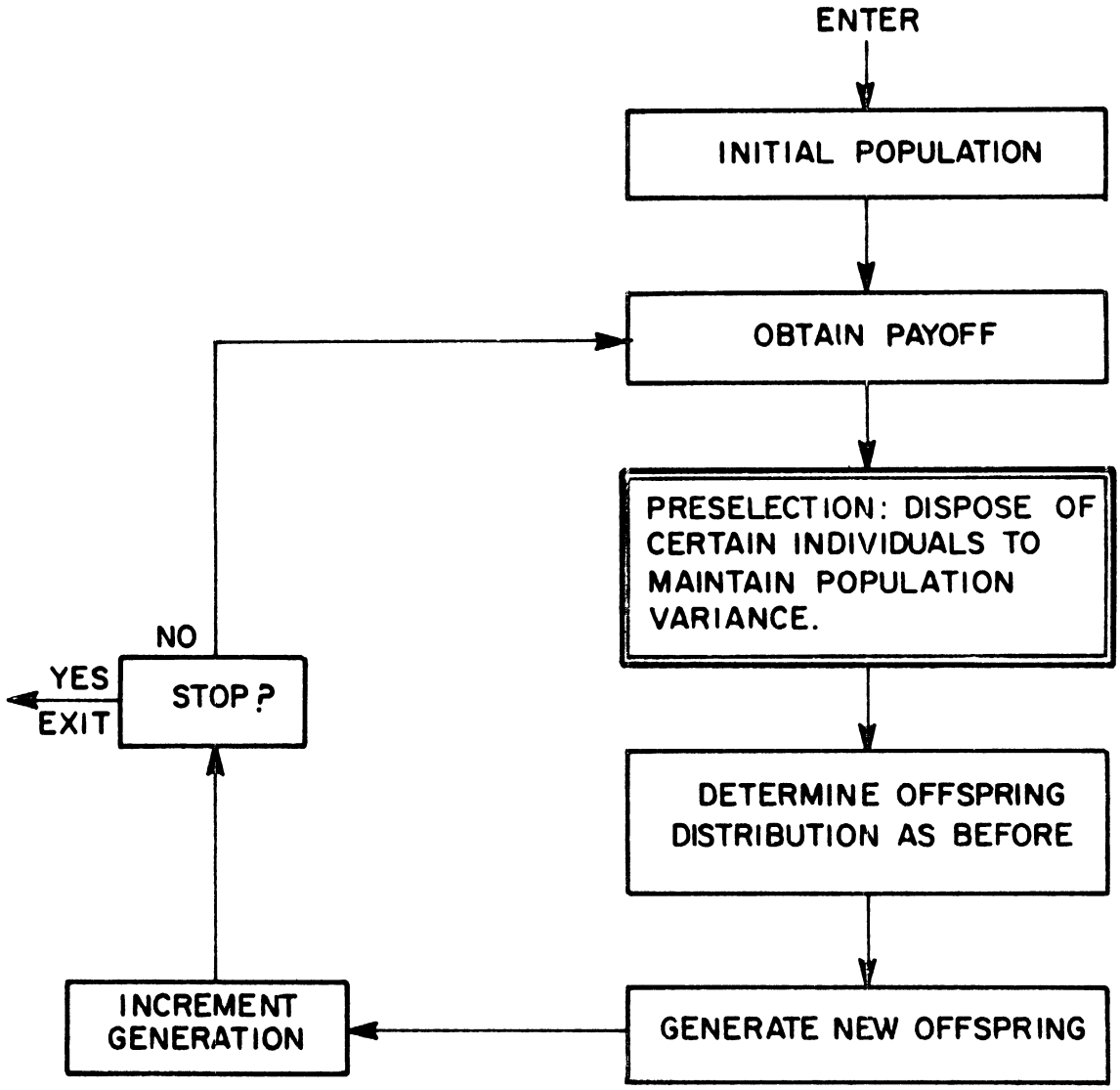


Figure 8.4 The introduction of preselection schemes to maintain population variance

to be discarded. However, this inferior offspring does resemble its parents and serves only to decrease population variance.

To eliminate this situation we check all offspring after they receive their payoff. If the offspring was created using one of the crossover operators, it must have performed better than *both* parents in order to assume a position in the current population. Otherwise it is discarded. If the offspring was formed using only the mutation operators, it must perform better than its parent or else it is discarded.

Performance with this scheme was a bit disappointing. Table 8.4, experiment 14, shows the performance at selected generations. When compared to experiment 17 (the previous constant chromosome scheme) it was significantly inferior at generation 150. This significance did not remain in later generations due to the large variance *between runs* of experiment 14. Furthermore judging from the spread of performance values in the population, it did not appear that experiment 14 really maintained any greater degree of population variance than did experiment 17. We may therefore conclude that this scheme is not a worthwhile addition to our plans.

Scheme 2

This scheme involved an addition to Scheme 1 which resulted in disposing of parent individuals when offspring were superior. If offspring were inferior to their parents, this scheme operated like Scheme 1. However, if an offspring was superior to its parents (or parent) then, in addition to its assuming a place in the current

EX NO	PSEL SCM	CONSTANT CHROM?	GEN 150	GEN 200	GEN 250	SIGN
14	1	YES	62.6	68.3	71.5	
15	2	YES	58.3	61.3	63.8	
16	3	YES	71.7	74.5	77.5	*
17	None	YES	69.9	72.3	75.5	
18	None	Ichrom. Dup.	76.8	79.2	79.8	

Table 8.4 Summary of Results Using Preselection Schemes

population, the worse parent (or only parent in the case of mutation) was discarded. As one can see this scheme puts a major emphasis on maintaining population variance. This was especially important in the case of an offspring formed using only the mutation operators, since in this case similarity is very high. With this scheme only the better of the parent and the offspring remains in the current population.

This scheme resulted in an extremely high amount of population variance. In fact, the best individual in the population typically received a payoff twice that of the *average* individual. However, the performance of the best individual was extremely poor when compared to other plans. Experiment 15 demonstrates this performance.

Scheme 3

It was surprising that Scheme 2 performed so poorly especially since it maintained a large population variance. One possible reason

is the rather strong requirements on offspring formed using crossover. About 75% of all new individuals in the first stages of adaptation are formed via crossover but very few performed better than both parents.

Scheme 3 modified Scheme 2 only in the case of offspring formed using crossover. In this case the offspring was required to perform better than only one parent in order to assume a place in the current population; in this case the inferior parent was discarded. If the offspring was worse than both parents, then it was discarded.

The improvement resulting from this scheme was enormous. Population variance was maintained to a greater extent than with experiment 17 (which used no preselection) and performance was higher. In fact, after 250 generations, experiment 16 significantly dominated experiment 17, and at this same point it was not significantly different from experiment 18 which was previously one of the best reproductive plans. Therefore, we may conclude that the third preselection scheme is a valuable addition to reproductive plans; it has improved the performance of the constant chromosome scheme and has maintained population variance.

Before ending this section we should comment on one bad aspect of Scheme 3 which may have prohibited it from performing even better in the late generations. Scheme 3 was developed after the parameter modification scheme but Scheme 3 definitely affects the behavior of the modification scheme since it plays a significant role in determining new population members. In one sense, it should have improved the parameter modification scheme since now the genetic operators are rewarded

based upon more valid information. In other words, now new population members are offspring which have truly performed better than their parents as a direct result of the genetic operators.

However, in another sense Scheme 3 seemed to adversely affect the parameter modification scheme since it favored new offspring created by crossover to a greater extent than new offspring created by mutation. As a result crossover rates rose to their maximum values so that every new individual was formed by crossover or double crossover. This situation is certainly not favorable, especially in the later stages of adaptation.

To avoid this situation we imposed limits of .6 on each crossover probability and a limit of .8 on the sum of the two probabilities. However, this did not significantly improve performance which already was high despite the high crossover rates. It is possible that the high crossover rates were not detrimental to adaptation; on the other hand, the imposed limits may still have been too high to affect any significant difference. In the final analysis we would suggest that the refinements of the parameter modification scheme mentioned at the end of Chapter 7 would most likely improve the performance of Scheme 3, but, nevertheless, Scheme 3 was very successful in maintaining population variance while increasing final performance levels.

8.5 Summary

In this chapter we have investigated a number of techniques in order to improve the performance of reproductive plans. Some have proven to be very beneficial while others did not significantly improve the performance of our plans.

Of greatest value was an evolutionary technique whereby the length of the chromosome increased as adaptation took place. Of three schemes investigated we found that two significantly improved the final performance of our reproductive plans. One scheme entailed evolving subpopulations of chromosome segments. After the performance gains of these populations leveled off the segments were concatenated to form normal sized chromosomes. The other scheme increased the chromosome length by duplicating some of the detectors already existent on the chromosome. This technique, which is analogous to an operation observed in nature, significantly improved both the efficiency and the effectiveness of the reproductive plan.

The second most successful scheme in this chapter involved the addition of a preselection technique which determined which individuals were permitted to become or remain members of the current population. This scheme provided a rather simple means for eliminating similar individuals without applying an elaborate similarity measure to all population members. As a result we are now able to maintain an evolving population of very different individuals with a resultant increase in performance.

The remaining techniques investigated in this chapter were generally

not as successful as the above. The mutation pool and dominance techniques upon which the pool is based show promise of becoming valuable additions to reproductive plans. However, our particular task did not seem to need such mechanisms. As a result performance increased only in a special case.

We also concluded that the technique of operating auxiliary populations did not prove to be a valuable addition. Our scheme for maintaining population variance has really eliminated much of the need for the new populations. However, the process of operating isolated populations and then mixing these at various stages of evolution may still prove valuable, especially in situations where the environment changes.

This chapter ends our experimental investigations into the behavior of reproductive plans. In the next and final chapter we will review some other work which has been done or will be done with reproductive plans, suggest some further additions to the reproductive algorithm, and then summarize our investigations.

Chapter 9 Overview and Conclusions

9.1 Other Uses of Reproductive Plans

Other researchers have used or are in the process of using reproductive plans of various types on various tasks. In this section we will briefly present some of this work to give the reader an idea of the breadth of possible applications of the reproductive paradigm and to point out some different implementations. This presentation will not examine any of the work in detail since much of it is still in the developmental stages. At the end of the section we will present some guidelines for further work with reproductive plans.

Current Research

The first major experimental investigation of reproductive plans was presented by Bagley [1967]. Bagley chose the game of hexapawn, a simple game played on a 3 x 3 board, for his task environment. This task was convenient for a number of reasons. Strategies for playing the game could be quantified into a set of parameter values analogous to Samuel's detector weights. The strategy of the opponent or, in other words, the environment, could be varied to make the play increasingly more difficult. Furthermore, this environment could be controlled so that performance would depend upon specific interactions between specific parameters. In addition, the entire playing process, player and opponent, could be simulated so that the cost of obtaining payoff was very small.

Bagley used diploidy (chromosome pairs) in order to include dom-

inance in his investigations. His plan operates as follows. Single chromosomes taken from a large population (200 chromosomes) are paired to create a diploid individual which defines a strategy for playing hexapawn. Dominance values assigned to the alleles determine which allele will be used in the strategy. After these chromosome pairs receive payoffs, each chromosome in the pair is duplicated a number of times according to how the performance of its pair compares to the average performance of the population. Then these duplicates are mated using crossover, inversion, and a single mutation operator.

Bagley compared his genetic plan to a correlation plan which involved correlating the success of the entire game-playing strategy with individual parameter values (and sometimes parameter pairs, triples, etc.). He found that the genetic adaptive plan performed very well when compared to the correlation plan especially in environments where payoff was affected by interactions between parameters.

One noteworthy result of Bagley's was the ineffectiveness of inversion. When a chromosome contains parameter values we must also encode into the chromosome indices to indicate which values correspond to which parameters; otherwise, an inversion would confuse the decoding process. A typical chromosome would consist of a string of 2-tuples as follows:

$$(p_1,1)\ (p_2,2)\ \ldots\ (p_i,i)\ \ldots$$

where p_i is the value for parameter i.

A problem will arise if crossover is attempted between two strings in which corresponding parameter values do not occur in the same positions on both strings due to previous inversions. If a crossover took place the result would possibly be a string having two values for some

parameters and no values for other parameters.

To avoid this problem Bagley only allowed crossover to take place between *homologous* regions of chromosome pairs, i.e., regions where the parameter indices matched position for position. However, after a number of generations, the chances of finding such a match between randomly selected pairs became very small. Therefore, inversion served only to delay adaptation since it restricted the use of the crossover operator.

Another significant feature of Bagley's plan is that the best individuals are not retained per se from one generation to another. An individual in this system is actually a pair of chromosomes, but each pair is split after each generation. It is difficult to say whether this hurt or helped Bagley's plan since he maintained relatively large populations (200-400 chromosomes) and the best *chromosomes* did remain from one generation to another. Therefore, there probably was a good chance that previous devices would be rediscovered due to duplication of chromosomes and dominance effects.

Weinberg also plans to use a reproductive plan to improve the performance of a biological model. His simulated cell of the bacterium *Escherichia coli* uses a number of parameter values to simulate enzyme characteristics under different environmental conditions [1969].

In Weinberg's reproductive plan the chromosome consists of a single string of parameter values and indices. A chromosome's utility is determined by how well the behavior of the simulated cell using the parameter values matches the behavior of real *E. coli* cells in similar environments. Weinberg's reproductive plan consists of operating four independent 10/6 populations similar to the populations we have used. He does not use diploidy as Bagley did and he saves the best individuals

after each generation.

During normal operation crossover and mutation (incrementing or decrementing parameter values by random amounts) are used to generate offspring for each isolated population. At specified generations during evolution the two worst populations are discarded and replaced by inverted copies of the two best populations. In other words, inversion is applied to all individuals in the population using the same breakage points so that all individuals in the new population are homologous. In this way, Weinberg can use inversion to modify linkage and thereby take advantage of interactions between parameters without affecting the operation of crossover.

Another interesting application of reproductive plans has been suggested by Goodman [1969]. The task is simply to find the maximum of a well-defined mathematical function. Therefore, the set of devices $\mathcal{A}$ is the set of points which define the domain of the function.

A chromosome consists of co-ordinate values suitably coded (e.g., using a binary representation) so that new values can be generated using genetic operators. In this representation one binary digit is analogous to an allele. One interesting aspect of Goodman's work was his demonstration that using only crossover certain coding procedures provide for more efficient adaptation than the standard binary encoding.

For example, the *Hamming distance* between two binary codes is defined as the number of positions in which the two codes do not match. Certain codes (called snake-in-the-box codes) have various distance-preserving properties meaning that the Hamming distance between two codes is related to varying degrees to the numerical distance between the values which the codes represent. These snake-in-the-box codes

prove to be better than a standard binary encoding for a number of interesting utility functions.

Guidelines for Further Work with Reproductive Plans

As one may suspect, the uses of reproductive plans is unlimited. All the researcher need supply is a chromosomal representation to effectively define the set of devices $\mathcal{A}$ and an accurate utility measure μ to rank the devices. Establishing a utility measure should present no problems as long as the researcher knows what kind of performance is desirable and as long as this performance measure can be easily obtained in a reasonable amount of time.

Finding a good representation for a device and/or generating the set $\mathcal{A}$ can pose more of a problem. The researcher must first decide upon which components of a device he desires to put under selection. In making this decision he must estimate, either intuitively or by obtaining payoff for a few devices, which components when changed will induce the greatest variability in performance. This is very important since variability is a crucial feature in the successful operation of reproductive plans.

After deciding upon which aspects of a device are to be varied during adaptation, the researcher must decide upon a suitable encoding in a chromosomal structure. This might involve deciding which parts will be analogous to genes to be exchanged using crossover and which parts, if any, will constitute intragene structure to be changed only by mutation. If mesh points were considered as genes in our representation, the concept of detectors as functional units would not be ap-

parent to the plan. If there appear to be additional functional units consisting of sets of genes, the researcher may wish to represent a device with a number of chromosomes. Such an extension of our reproductive plan should be fairly straightforward.

After settling upon an adequate representation the researcher may have to exercise additional caution in constructing suitable mixing operators. In fact, his awareness of the problems involved in the effective operation of genetic operators may very well influence his choice of a representation. Mutation-like operators typically will not cause any problems. The researcher should try to have a variety of these operators to enable the plan to take different sized steps in the search space and he should use his intuition and any previous knowledge about the task in developing meaningful mutation operators.

Crossover-like operators may be more difficult to use successfully. As we have seen in Bagley's work, the simultaneous use of crossover and inversion is not always easy to implement. Furthermore, there are a number of representations which may not be preserved under the operation of crossover. In other words, the offspring created by crossover may not be codes for any device. In cases such as these, one must be careful to use mixing operators which are closed in the set of permissable chromosomal codes.

Once he has decided upon a suitable representation, utility measure, and set of genetic or mixing operators, the researcher can proceed to operate a reproductive plan. Hopefully at this point he can draw upon many of the techniques that we have investigated in this work in order to provide for efficient and effective operation of his reproductive

plan. With proper care this procedure should result in meaningful extensions of much of the work done in artificial intelligence.

9.2 Extensions of the Reproductive Paradigm

Throughout our work we have touched upon only a small number of possible reproductive plans. In the previous section we reviewed some previous and ongoing research which is investigating other implementations of the general reproductive paradigm. Now we would like to suggest a few extensions of the plans we have been working with in hopes of spurring continued interest in the study of reproductive adaptive plans.

We have already suggested possible extensions and improvements of the schemes we have investigated. Near the end of Section 7.2 we presented a critique of the parameter modification scheme. In Section 8.2 we suggested how the use of isolated and new populations might be improved. In Section 8.3 we presented the dominance effect as a potentially valuable addition to reproductive plans and proposed how our mutation operators might be modified to increase the chances of finding valuable new detectors. Finally, at the end of Section 8.4 we suggested how our preselection scheme might be improved in conjunction with an improved parameter modification scheme so as to maintain population variance.

One possible extension that we have not mentioned involves the use of additional feedback. From our analysis of other adaptive systems in Section 2.2 we may conclude that researchers generally feel that they can extract additional feedback information for use in an

adaptive plan. As we have mentioned previously, exclusive use of this feedback, which usually takes the form of a component evaluation, may result in a very inflexible adaptive plan which searches only a small portion of the set $\mathcal{A}$. Yet, this additional feedback may prove valuable to a reproductive plan by directing many of the processes which, in our work, have been completely random.

However, we must stress that in the final analysis the cost of this additional feedback must be weighed against the potential or actual gain. If the extraction of the additional feedback takes again as much time as the extraction of the overall utility μ, we will expect reproductive plans using the additional feedback to require only half the number of samples to discover peaks similar to those discovered by a plan not using the additional feedback. Of course, if the plan which does not use the additional feedback is incapable of discovering devices of the same quality as those discovered by plans using the additional feedback, then the extra cost is certainly justifiable.

Let us now look at some possible uses for this additional feedback. Suppose we had some information as to which detectors or components were possibly inferior to the rest. This is the type of information often used by other adaptive plans. We could use this information to generate a nonuniform distribution over the components of each device. This distribution could be used by the mutation operators so that the inferior components are more likely to be mutated while the superior components are less likely. We should stress, however, the importance of maintaining the probabilistic nature of the plan rather than directing mutation towards the worst component as is often done. As we mentioned before, the additional information is usually not completely

valid either because it is the result of a poor measure or because its use is based upon independence assumptions which do not hold. The more valid we feel the information is, the more we may bias our distribution.

We may also have some information that indicates *why* a particular detector or component is inferior. This information could be used to decide which mutation operator should be applied and even how it should be applied. In other words, we may be able to bias the choice of a detector or component.

The operation of crossover may also benefit from additional information. Crossover operations between "similar" individuals are more likely to result in superior offspring than crossover between dissimilar individuals. However, similarity can not always be defined in terms of utility. Additional information might identify such similarities and might also identify individuals which perform complementary roles in executing the task. In this latter case, there may be a good chance that offspring created by crossover will be able to perform both roles, especially if there is redundancy in the chromosome.

Any information concerning the interaction between components could be used to bias the choice of breakage points for the operations of crossover and inversion. Components which interact in a favorable manner should not be separated by crossover and should become highly linked using inversion.

We should mention that most of the operations in our plan which were completely random are generally not random in natural genetic systems. Different genes have different susceptibility to mutation, certain types of mutation are more probable than others, breakage is not

equiprobable at each possible breakage point, and individuals do not mate randomly. Many of these operations presumably have regulatory sites which themselves come under selection. Rosenberg [1967] has investigated the use of such regulatory sites in directing breakage for the crossover operation. Similar procedures could be used to improve other probabilistic aspects of the reproductive plan which we have made completely random.

9.3 Conclusions

Formal Summary

Let us now use our formal framework to briefly summarize our experiments with reproductive plans. In all the experiments the set $\mathcal{A}$ was the set of detector strings of a certain specified length. A few of our schemes, however, used only portions of the detector strings. The variable chromosome schemes effectively began their search in small spaces by using only a few of the available detectors on the string. When this chromosome segment seemed unable to continue substantial performance growth, it was extended to obtain an increased information capacity. The search eventually took place in the full space $\mathcal{A}$.

The set $\mathcal{E}$ consisted of the different pattern recognition tasks. Two task environments (called the easy and difficult tasks) were used for the majority of the experiments. A criterion based upon the best device obtained after a certain number of time steps was used for most of the experiments. In these cases runs were continued until a fixed number of generations elapsed or until the marginal utility (slope of the performance curve) fell below a certain level. Many experiments were evaluated with respect to both of these stopping rules.

The majority of the experiments were concerned with searching or generating the set $\mathcal{T}$. The feedback $\mathcal{J}$ consisted only of the payoff $\mu(A)$ for each device. The current memory M_t, the device selection function $\tilde{\tau}$, and the memory updating function m varied over the different experiments. At the beginning the memory state M_t consisted only of the payoffs for each device in the current set. The device selection function $\tilde{\tau}$ involved the basic reproduction and recombination steps. Table 9.3.1 briefly summarizes the different schemes we have investigated and indicates how these schemes enlarged the set $\mathcal{T}$ through changes in M_t, $\tilde{\tau}$, and m. Changes in M_t should be interpreted as additions in most cases.

Overview

The goal of this dissertation has been to develop a good set of adaptive plans for use in artificial intelligence work. As with many goals of this type, it was beneficial to restate the goal in many different ways so as to beter our understanding of the problems involved. Therefore, we first tried to answer the more basic questions : "What is an adaptive plan?", "What constitutes a *good* adaptive plan?", and "What difficulties might keep an adaptive plan from becoming good?" To answer these questions we developed a formal framework with which we could analyze adaptive systems and applied this framework to some current adaptive systems. As a result we extracted some of the difficulties that these systems encountered and gained a better understanding of what "good" means with respect to adaptive plans. At that point our goal became more concrete.

Many of the adaptive plans that we analyzed encountered similar

Scheme	Modification Involved		
	M_t	$\tilde{\tau}$	m
Basic	Payoffs of current population	Reproduction and recombination	Save payoff of new individuals
Offspring Selection	Total number of offspring over generations	Change reproduction (Sampling scheme) for a more optimal balance	Update total number of offspring
Parameter Modification	1) Current set of parameters 2) Current observed frequency of operator usage in new population members 3) Number of new population members since last change in parameter settings	Uses *current* parameters for recombination	Update current parameters (various schemes were tried to reduce fluctuations and produce consistent performance)

Table 9.3.1 Summary of Schemes Interpreted in the Formal Framework
(Continued on next page)

Scheme	Modification Involved		
	M_t	$\tilde{\tau}$	m
Variable length chromosomes		Searches a smaller space initially	
Isolated populations	Extra population	Independent populations (searches) which are later combined	
New populations	Auxiliary population	replaces current population with "spare" when performance levels off	Updates auxiliary population by randomly replacing auxiliary population members with current population members
Mutation pool	Auxiliary pool of alleles for mutation II	Mutation II picks from pool	Update pool last in, first out
Preselection to maintain population variance		Broader sampling possibilities by eliminating similar individuals	

Table 9.3.1 Summary of Schemes Interpreted in the Formal Framework (Concluded)

difficulties. Basically these difficulties stemmed from the improper use of feedback information, although poor representation also played a significant role. As a result many of these plans, in our opinion, were capable of carrying on only a limited search of the space $\mathcal{A}$. Additional inflexibility also resulted from biases usually based upon the researcher's incorrect assumptions about the task and the devices.

We decided to investigate a class of adaptive plans which showed promise of performing very well in situations in which the adaptive plans that we analyzed would probably perform very poorly. This basically is a situation in which we have little specific knowledge about the task and the devices but in which we can estimate that there may be strong interactions between components of the devices. These plans are called genetic or reproductive adaptive plans.

On the one hand, Holland's theoretical results indicated that this class of plans is optimal with respect to an appropriate quantitative definition. On the other hand, Bagley's empirical results showed that a more specific set of these plans performed as well as other widely used adaptive plans (correlation plans). The class of reproductive plans, however, is very large. Our goal, therefore, was reduced to finding a good subset of reproductive plans.

We claim to have achieved this goal. Guided by the maximum utility criterion mentioned above, we have developed a subset of reproductive plans what are much better than the detector evaluation plans and which are better than other implementations of the reproductive paradigm. We feel this is a major result especially since the criterion takes cost considerations into account.

The reproductive plans that we have developed have proved to operate

both efficiently and effectively on our pattern recognition task. We have mentioned in several places how the successful operation of these plans depends upon the interaction of many of the schemes we have developed. For example, we have demonstrated that a large member population is necessary for the successful operation of the parameter modification scheme, for efficient use of the crossover and inversion operators, and for efficient use of our intrachromosomal duplication scheme, which resulted in one of the best plans. Furthermore, we have seen that this large member population would be of no benefit without a selection scheme that utilized many of the population members. In addition, we have suggested how the preselections schemes, which maintain population variance, might be improved by improving the parameter modification scheme. As a further check on the goodness of many of the schemes we developed, we have operated one of our best reproductive plans in a number of testing situations, each time replacing a different part of this very good plan with an alternative scheme. In each case, the result was significantly inferior performance.

To sum up how our reproductive plans have improved the performance of the pattern recognition paradigm, we have plotted in Figure 9.3 some representative runs of the detector evaluation plan (called the control experiment) and two reproductive plans operating on the difficult task. We have plotted "Number of devices tested" on the abscissa rather than number of generations since the detector evaluation plan samples only one device per time step. The reproductive plans do not improve initially due to the sampling of the initial population. Performance for the reproductive plans is average performance of the current population.

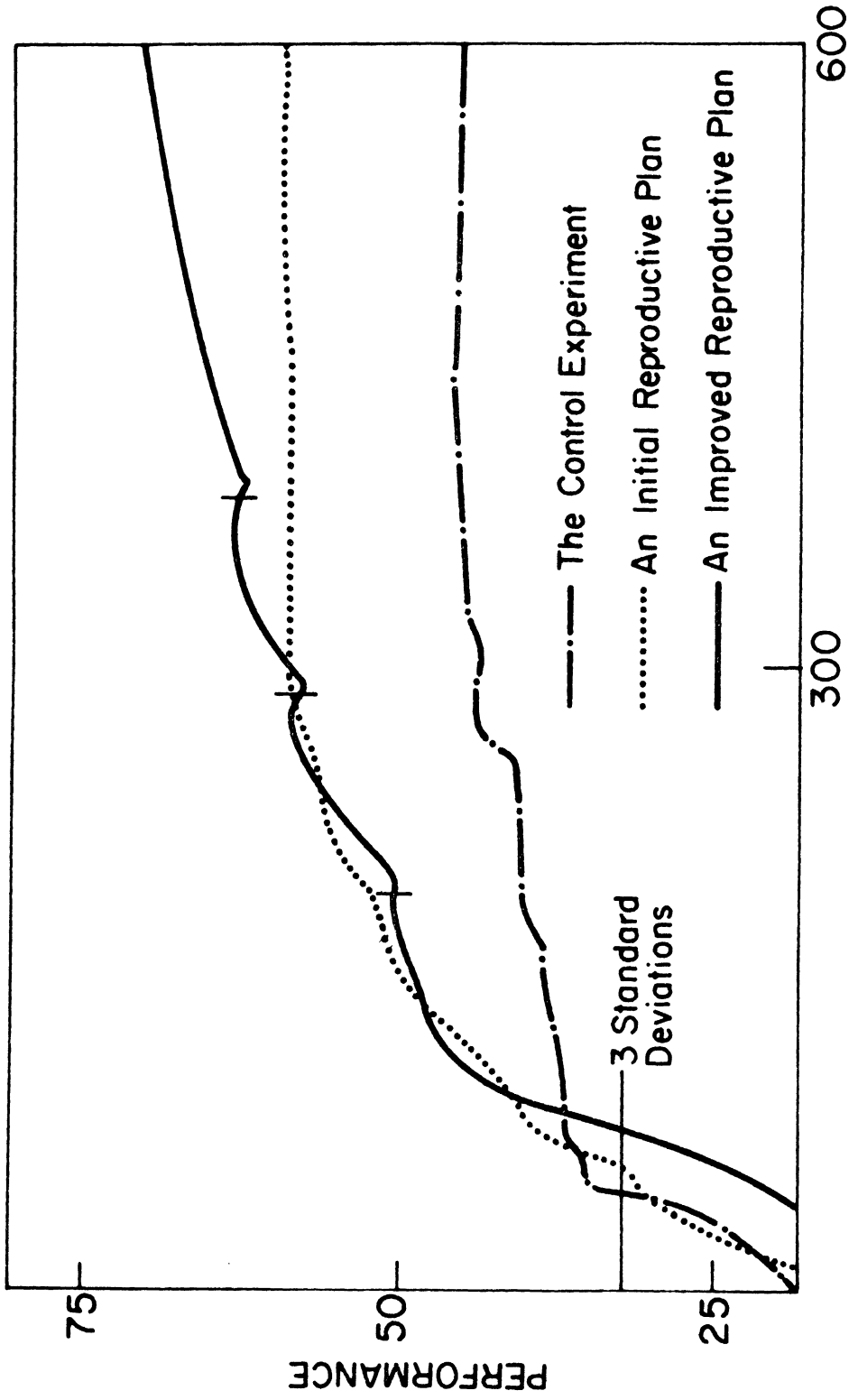


Figure 9.3 Comparison of three different adaptive plans

We have also indicated the three-standard-deviations level of the random plan.

As one may recall, the detector evaluation plan operated in a serial manner (one device generated and saved each time step). Detectors were evaluated and the worst ones replaced using our mutation operators. The run that is plotted is actually better than the average run. However, all runs exhibited a leveling off trend after sampling about 300 devices.

The "initial" reproductive plan is representative of runs after different selection schemes had been investigated but before parameter modification schemes had been introduced. It too levels off after 300 devices but at a much higher level than the detector evaluation plan. This leveling off might be attributed to the plan's inability to take smaller steps in the search space (i.e., due to the lack of the parameter modification scheme).

The "improved" reproductive plan represents one of the best plans we have developed. This plan used the best parameter modification scheme, the interchromosomal duplication scheme, and maintained a large population (34 individuals) with a small number of samples per generation (6). The vertical lines along the performance curve indicate points at which the chromosomes were extended. These improved reproductive plans did not usually level off until performance was above the 75 level. We should note that the control experiment took twice as long as the reproductive plans to obtain payoff due to the additional feedback required. This time difference is not reflected in Figure 9.3.

Our reproductive plans are also good for other reasons not included in the criterion χ. With the successful implementation of the

operator parameter modification scheme, we have produced plans which will remain good over different environments, representations, and specifications of mixing operators. In other words, we have added an element of flexibility.

Finally, our plans are good in that they were not restricted by the assumptions that other systems made (such as, independence of components). Therefore, the plans were not limited to any particular type of search.

In this study we have investigated a number of different schemes; some have proved valuable while others have not. However, all were investigated with specific purposes in mind. In Table 9.3.2 we have summarized these schemes indicating the purposes for investigating them along with an estimate of their value or importance to reproductive plans in general.

As with most work of this nature, we were not able to thoroughly investigate each scheme. Hopefully some of these schemes will be tried again using different tasks and different implementations of the reproductive plan. In particular one should try to operate reproductive plans with more than payoff-only feedback, as we have suggested in the previous section. Using this feedback, one would be in a better position to accurately compare reproductive plans to nonreproductive plans which use the same representation.

However, accurate comparisons between a wide variety of adaptive plans and learning techniques will only be possible when we have available a number of standardized tasks upon which a number of plans can operate. We feel we have made progress in resolving some of the problems involved with evaluating different adaptive plans by considering a number of alternative plans and using quantitative criteria to measure a plan's success. However, much more work is needed along these lines.

Scheme	Effect on Performance	Purpose for Trying this Scheme	Estimated Value or Importance of Scheme on Other Tasks
Offspring Selection	Significant improvement	1) To improve sampling techniques in the search space. 2) To maintain population variance.	Very important
Probability Modification Scheme	Significant improvement	1) To satisfy different search needs at different points in adaptation and on different tasks. 2) To relieve researcher of problem of finding best settings and evaluating individual operators.	Very important to all reproductive plans expecially when working with a new task.
Variable Length Chromosomes: Random extension Concatenation Intrachromosomal duplication	 No difference Significant improvement Significant improvement	1) To add redundancy to chromosome. 2) To force device to use limited amount of resources early in adaptation. 3) To achieve steady increase in performance and ultimately higher peaks.	Random: of little value. Concatenation: may continue to be valuable in a representation where each segment has a different function. Intrachromosomal duplication: may continue to be important if representation allows for different size chromosomes.

Table 9.3.2 Summary of Scheme's Performance and Estimated Value
(Continued on next page)

Scheme	Effect on Performance	Purpose for Trying this Scheme	Estimated Value or Importance of Scheme on Other Tasks
Isolated Subpopulations	No difference	To reach higher peaks by maintaining a parallel search.	May prove helpful with suggested modifications or when using other representations (e.g., See Weinberg)
New populations	No difference	To provide a new population mix when the current population performance levels off.	Probably not important with larger populations or preselection schemes which maintain variance.
Mutation Pool	Improvement when using mutation alone	To provide rapid access to valuable alleles.	Probably very important with varying environments and when certain alleles are much more valuable than others.
Preselection to maintain variance	Significant improvement	To maintain a rich mixture of individuals in the population.	Very important for all reproductive plans since insufficient population variance can seriously inhibit adaptation.

Table 9.3.2 Summary of Scheme's Performance and Estimated Value
(Concluded)

REFERENCES

Bagley, J. D. (1967) "The Behavior of Adaptive Systems Which Employ Genetic and Correlation Algorithms," Technical Report 01252-1-T, Computer and Communication Sciences Department, The University of Michigan.

Bledsoe, W. W. and Browning, I. (1959) "Pattern Recognition and Reading by Machine," in *1959 Proceedings of The Eastern Joint Computer Conference*, pp. 225-232.

Crow, J. F. and Kimura, M. (1970) *An Introduction to Population Genetics Theory*, New York: Harper and Row.

Feigenbaum, E. A. and Feldman, J. (1963) *Computers and Thought*, New York: McGraw-Hill Book Co.

Fogel, L. J.; Owen, A. J.; and Walsh, M. F. (1966) *Artificial Intelligence Through Simulated Evolution*, New York: Wiley.

Friedberg, R. M. (1958) "A Learning Machine, Part I," *IBM Journal*, January.

Goodman, E. D. (1969) "Maximizing Via Reproductive Plans: Some Informal Considerations," Paper presented in the course *The Theory of Adaptive Systems*, Computer and Communication Sciences Department, The University of Michigan.

Helling, R. (1968) "Selection of a Mutant of *Escherichia Coli* Which has High Mutation Rates," *J. Bacteriol.*, *96*, pp. 975-980.

Hoel, P. G. (1963) *Introduction to Mathematical Statistics*, New York: Wiley.

Holland, J. H. (1967) "Nonlinear Environments Permitting Efficient Adaptation," in *Computer and Information Sciences-II*, New York: Academic Press Inc.

___, (1969) "Adaptive Plans Optimal for Payoff-Only Environments," in *Proceedings of the Second Hawaii Conference on System Sciences.*

___, Forthcoming book on adaptive systems tentatively titled: *Adaptation in Natural and Artificial Systems.*

Hubel, D. H. and Wiesel, T. N. (1959) "Receptive Fields of Single Neurons in the Cat's Striate Cortex," *Journal of Physiology*, *148*, pp. 574-591.

Klopf, A. H. (1965) "Evolutionary Pattern Recognition Systems," Report from the Bioengineering Section, Information Engineering Department, The University of Illinois, Chicago.

Lin, S. (1968) "Heuristics for Solving Large Combinatorial Problems on a Computer," talk given at *Fourth Systems Symposium*, Case Western Reserve University.

Lindsay, R. K. (1968) "Artificial Evolution of Intelligence," *Contemporary Psychology, 13*, No. 3, March.

Newell, A. and Simon, H. A. (1963) "GPS, A Program that Simulates Human Thought," in Feigenbaum and Feldman's *Computers and Thought (1963)*.

Newell, A.; Shaw, J. C.; and Simon, H. A. (1960) "A Variety of Intelligent Learning in a General Problem Solver," in Yovitts, M. and Cameron, S. (eds) *Self-Organizing Systems*, New York: Pergamon.

Rosenberg, R. (1967) "Simulation of Genetic Populations with Biochemical Properties," Technical Report 08333-2-T, Computer and Communication Sciences Department, The University of Michigan.

Samuel, A. L. (1963) "Some Studies in Machine Learning Using the Game of Checkers," in Feigenbaum and Feldman's *Computers and Thought (1963)*.

___, (1967) "Some Studies in Machine Learning Using the Game of Checkers II-Recent Progress," *IBM Journal*, November.

Uhr, L. and Vossler, C. (1963) "A Pattern Recognition Program that Generates, Evaluates, and Adjusts its own Operators," in Feigenbaum and Feldman's *Computers and Thought (1963)*.

Wallace, B. (1968) *Topics in Population Genetics*, New York: Norton.

Weinberg, R. (1970) "Computer Simulation of a Living Cell," Ph.D. Dissertation, Computer and Communication Sciences Department, The University of Michigan.