

AD A101476

AFWAL-TR-81-1070



GOAL SEEKING COMPONENTS FOR ADAPTIVE INTELLIGENCE: AN INITIAL ASSESSMENT

DTIC
ELECTE
S JUL 16 1981 D
E

Computer and Information Science
University of Massachusetts
Amherst, MA 01003

April, 1981

TECHNICAL REPORT AFWAL-TR-81-1070

Final Report for Period 1 September 1977 to 31 August 1980

Approved for public release; distribution unlimited

AVIONICS LABORATORY
AIR FORCE WRIGHT AERONAUTICAL LABORATORIES
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433

81 7 14 009

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This report has been reviewed by the Office of Public Affairs (ASD/PA) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

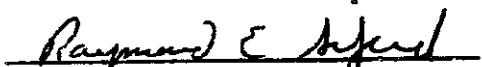


A. HARRY KLOFF
Project Engineer



DONALD L. MOON
Chief, Information Processing
Technology Branch
System Avionics Division

FOR THE COMMANDER



RAYMOND E. SIFERD, COLONEL, USAF
Chief, System Avionics Division

"If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify AFVAL/AAT, W-PAFB, OH 45433 to help us maintain a current mailing list".

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

DD FORM 1473 EDITION OF 1 NOV 65 IS OBSOLETE
1 JAN 73

SECURITY CLASSIFICATION OF THIS PAGE (AND PAGE REVERSE)

20. Abstract

having these capabilities have not been studied previously. We demonstrate that simple networks of these elements can solve types of problems that are beyond the capabilities of networks studied in the past. An associative memory is presented that retains the generalization capabilities and noise resistance of associative memories previously studied but does not require a "teacher" to provide the desired associations. It conducts active, closed-loop searches for the most rewarding associations. We provide an example in which these searches are conducted through the system's external environment and an example in which they are conducted through an internal predictive model of that environment. The latter system is capable of a simple form of latent learning. We argue that components capable of making progress toward their goals when embedded in environments that are indifferent, or even hostile, with respect to these goals can form the substrate for a decentralized, parallel, and pervasively adaptive problem solver. We discuss the hypothesis that neural information processing can be understood in these terms, and we relate our results to animal learning data.

Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution:	
Availability Codes	
Dist	Avail and/or Special
A	

FOREWORD

This report describes research supported by the Air Force Office of Scientific Research and the Avionics Laboratory (Air Force Wright Aeronautical Laboratories) through contract F33615-77-C-1191 Project 2304, "Adaptive Network Simulations." The work reported here was performed during the period 1 September 1977 through 31 August 1980 under the direction of Principal Investigators D. N. Spinelli, W. L. Kilmer, and M. A. Arbib. The report was prepared by A. G. Barto and R. S. Sutton and released in February 1981.

The objective of this contract was to study the feasibility of using goal-seeking elements as components of machines capable of achieving intelligent, goal-directed performance such as image understanding, speech recognition and decision making. As such, much of the work is ground-breaking and exploratory. The three main aspects of the accomplishments are: 1) the implementation of graphic and programming tools for computer simulation; these are as important to us as a particle accelerator is to a nuclear physicist, 2) the experimental study of a great variety of goal-seeking components analysed as isolated elements. This mapping of rules for adaptive goal-seeking has made clear what will and will not work and why, pointing the way to

further optimization of the elements. 3) The demonstration in principle that small goal-seeking nets could be built out of components that are themselves goal-seeking, and further, that such nets are capable of substantive adaptive behaviors.

We conclude that goal-seeking elements have unusual power as components of goal-seeking nets. In fact, it seems highly probable that only by using such components will we be able to understand those sentient attributes of intelligence, such as image understanding, that have proved so resistent to Artificial Intelligence methodology. The task of assembling adaptive components into sophisticated structures capable of truly complex performance in terms of vision, speech and decision support remains for the future.

The Department of Computer and Information Science at the University of Massachusetts in Amherst has provided a unique environment for this project. State of the art research in Artificial Intelligence, Cybernetics, Computer Systems, and Natural Intelligence is ongoing in very strong groups whose expertise we continually partake of. To them and all others who have helped, our thanks.

D. N. Spinelli, Principal Investigator
Professor of Computer Science
Univ. of Mass., Amherst, MA
Nov. 7, 1980

Acknowledgements

As the principal authors of this report, we would like to express our gratitude to the many people who have made this effort possible, encouraged us, and from whom we have learned a great deal. First, our deepest thanks to Harry Klopf for bringing to us a set of ideas so rich in possibilities. We thank the faculty of the Computer and Information Science Department at the University of Massachusetts at Amherst for creating an exceedingly stimulating atmosphere. Through them, we have maintained contact with state of the art developments in a diversity of areas and have had access to state of the art computational tools. We especially thank William Kilmer, Michael Arbib, and Nico Spinelli, who have most directly guided our efforts and whose unmatched contributions to theoretical and experimental neuroscience have strongly influenced this research.

To Oliver Selfridge, John Moore, William Hodos, Chuck Woody, Hewitt Crane, and John Holland we are very grateful for permitting us to benefit from their wealth of experience, expertise, and scholarship.

We thank Peter Brouwer, Chuck Anderson, Eva Hudlicka, Jack Porterfield, Stephen Epstein, Daryl Lawton, Gonzalo

Viana De Prisco, Michael Poe, Rolando Lara, Jeff Mischkinsky, and George Peredy for contributing to this research in important ways: for pointing us toward useful areas of literature, for helping to perform the many computational experiments this research has entailed, and for providing fresh ideas and stimulating discussion. We especially thank Peter Brouwer who developed the simulations described in Section 6, and Chuck Anderson who developed a large part of our simulation software. We also thank Susan Parker for applying her managerial skills to many aspects of this project.

Finally, of course, we must claim for ourselves any fallacies or naivete~ that are revealed in this report.

Andrew G. Barto

Richard S. Sutton

Amherst
November, 1980

TABLE OF CONTENTS

Section 1. INTRODUCTION

Section 2. ADAPTIVE SYSTEM THEORY

2.1 Introduction	2-1
2.2 Problems Versus Mechanisms	2-4
2.3 Basic Adaptation and Learning Problems	2-7
2.3.1 Some Basic Distinctions	2-12
2.3.1.1 Nature of Control Over Input	2-12
2.3.1.2 Number of Control Situations	2-15
2.3.1.3 Control Surface Knowledge	2-16
2.3.1.4 Knowledge of Preferences	2-17
2.4 Particular Problems and Methods	2-23
2.4.1 Pattern Recognition Problems	2-23
2.4.2 Clustering	2-31
2.4.3 Stochastic Approximation Methods	2-32
2.4.4 The Perceptron Learning Rule	2-34
2.4.5 Function Optimization	2-38
2.4.6 Learning Automata	2-42
2.4.7 Closed-Loop Control	2-47

2.4.8 Kineses and Taxes	2-50
2.4.8.1 Kino-Kinesis	2-51
2.4.8.2 Tropo-Taxis	2-56
2.4.9 Adaptive Control	2-58
2.4.10 Learning Control	2-64
2.4.11 Instrumental Conditioning	2-70
2.4.12 Klopf's Heterostat	2-74
2.5 Summary and Discussion	2-73

Section 3. EVOLUTION OF HETEROSTAT MODELS

3.1 Introduction	3-1
3.2 Early Models: Open-Loop Stability	3-2
3.2.1 Weighted Correlation Element with Zerosetting	3-3
3.2.2 Open-Loop Stability	3-8
3.2.3 Zerosetting	3-11
3.2.4 Change In Input as Reinforcement	3-12
3.2.5 The \dot{y} Element	3-15
3.3 Other Elements using Change In Input as Reinforcement	3-17
3.3.1 The Exponential Trace Eligibility \dot{y} Element	3-17
3.3.2 The \dot{s} element	3-21
3.3.3 The "Dual" Heterostat	3-22
3.3.4 The Classical Conditioning Predictor Element	3-23
3.3.5 Dotting the x Eligibility Term	3-24
3.3.6 Dotting the y Eligibility Term	3-26
3.3.7 Separate x and y Averages in Eligibility	3-27

3.3.3 Problems with Models using the Change In Input as Reinforcement	3-28
3.3.3.1 The End Reinforcement Problem	3-28
3.3.3.2 Conflict between the Selecting and Reinforcing Functions of Input	3-32
3.4 Models with Specialized Reinforcing Input Lines	3-33
3.4.1 ALOPEX as an Action Selector	3-34
3.4.2 Associative Search Network Element	3-34
3.4.3 The Associative Search Problem	3-35
3.4.3.1 Null Transitions	3-36
3.4.3.2 Associative Search with Predictor Element	3-38
3.4.3.3 Learning Situation Transitions	3-40
3.4.4 An Element that Makes Two Uses of Prediction	3-43
3.4.5 A Proposal for an Alternative Problem	3-44
3.4.5.1 The Problem Schema	3-47
3.4.5.2 Payoff Functions	3-49

Section 4. OPEN-LOOP LEARNING: EXPECTATION, PREDICTION,
AND CLASSICAL CONDITIONING

4.1 Introduction	4-1
4.2 Adaptive Element Analogs of Classical Conditioning	4-8
4.3 Temporal Relationships	4-19
4.3.1 Delays	4-25
4.3.2 Stimulus Traces	4-28
4.3.3 Non-Stimulating Traces	4-31
4.3.4 Model Behavior in Classical Conditioning with a single CS	4-35

4.4 Context and Expectation	4-44
4.4.1 Higher Order Conditioning	4-59
4.5 Adaptive System Theory	4-61
4.5.1 Hebbian Rule	4-62
4.5.2 Widrow-Hoff Rule	4-65
4.5.3 Rescorla-Wagner/Widrow-Hoff Predictor	4-71
4.5.4 Uttley's Informon	4-74
4.5.5 Our Model	4-77
4.6 Stability and Saturation	4-78
4.6.1 Normalization	4-81
4.6.2 Autonomous Decay	4-85
4.6.3 Negative Feedback	4-86
4.7 Cellular Mechanisms	4-90
4.8 Summary and Conclusions	4-98

**Section 5. ASSOCIATIVE SEARCH NETWORK: A REINFORCEMENT
LEARNING ASSOCIATIVE MEMORY**

5.1 Introduction	5-1
5.2 The Associative Search Problem	5-6
5.3 The Basic Adaptive Element	5-8
5.4 The Problem of Context Transitions	5-13
5.5 A Network	5-16
5.6 Examples	5-20
5.7 Neural Search	5-32
5.8 Sensorimotor Control Surfaces	5-35
5.9 Conclusion	5-37

**Section 6. LANDMARK LEARNING: AN ILLUSTRATION OF
ASSOCIATIVE SEARCH**

6.1	Introduction	6-1
6.2	Associative Search	6-2
6.3	Spatial Learning as Associative Search	6-3
6.4	The Learning Rule	6-8
6.5	Learning in a Noiseless Environment	6-11
6.6	Relearning in a Modified Environment	6-18
6.7	Learning in a Noisy Environment	6-20
6.8	A Remark on Linearity	6-23
6.9	Conclusion	6-24

**Section 7. AN ADAPTIVE NETWORK THAT CONSTRUCTS AND USES
AN INTERNAL MODEL OF ITS ENVIRONMENT**

7.1	Internal Models for Search and Simulation	7-1
7.2	The Simulated Task Environment	7-12
7.3	The Simulated Adaptive Network	7-15
7.3.1	The Action Selecting Component	7-17
7.3.2	The Internal Model Component	7-22
7.4	The Exploration Phase	7-28
7.5	The Association Phase	7-30
7.6	The Representation Problem	7-31
7.7	The Testing Phase	7-35
7.8	Superstitious Learning	7-36
7.9	Discussion	7-42

Section 8. GOAL-SEEKING SYSTEMS OF GOAL-SEEKING COMPONENTS

8.1 Introduction	8-1
8.2 Goal-Seeking Components	8-2
8.2.1 Learning with a Teacher	8-3
8.2.2 Learning without a Teacher	8-5
8.2.3 Learning with a Critic	8-6
8.2.4 Learning with an Occasional Critic	8-9
8.2.5 Nature as an Occasional Critic	8-11
8.3 Networks of Goal-Seeking Components	8-12
8.4 Games and Cooperation	8-15
8.4.1 Group Optimality	8-17
8.4.2 Goal-Seeking Systems as Game Players	8-21
8.5 Coalitions and Cell Assemblies	8-23
8.6 Heterostats as Cooperating Game Players	8-27
8.7 Cooperation by the Creation of Environments	8-29
8.8 Control Strategies for Problem Solving	8-31
8.9 Neurons as Goal-Seeking Systems	8-34
8.10 A Sense of Neural Function	8-39

Section 9. CONCLUSIONS

9.1 What is New?	9-2
9.2 Open-Loop Learning	9-7
9.3 Generalized Reinforcement	9-9
9.4 Associative Search	9-11
9.5 Biological Implications	9-12
9.6 In Summary	9-15

Appendix A. ANALYSIS OF STEADY STATE BEHAVIOR OF THE
RESCORLA-WAGNER/WIDROW-HOFF PREDICTOR FOR
A SIMPLE CASE

Appendix B. A FORMAL DESCRIPTION OF THE MODEL SIMULATED
IN SECTION 4

Appendix C. ADAPTATION OF LEARNING RATE PARAMETERS

Appendix D. DETAILS OF THE SIMULATION EXPERIMENTS
OF SECTION 7

Appendix E. DETAILS OF THE ADAPTIVE NETWORK SIMULATED
IN SECTION 7

Appendix F. DESIGNNET: NETWORK SIMULATION DISPLAY PACKAGE

Appendix G. EXPERIMENTER: SIMULATION OF A ROBOT'S
ENVIRONMENT

Bibliography

SECTION 1

INTRODUCTION

It is traditional that man's most complex invention at any time in history becomes a metaphor for neural function. When the early computer studies of hypothetical neural networks were undertaken, this role was filled by the electronic digital computer itself. The view of a brain as a computer was naturally accompanied by the association of the neuron, usually considered to be the basic functional component of nervous systems, with such computer components as logic gates, transistors, or other devices that were in themselves among the genuinely remarkable new inventions of those times. Now, due to the extraordinary progress in integrated circuit technology, the digital computer has become conceptually manageable, shrinking in size and becoming a familiar part of our lives, so that we have little trouble thinking of an entire computer as a "primitive" component of an even more complex system - a distributed processing system. Among the profound social consequences of this technology will be its influence on our metaphors for neural function. Even using considerable

imagination it is difficult to think of a room-sized machine as a primitive component. That this same amount of machinery can now be held in one's hand, however, sets the stage for an entirely new round of neural theory. This time it is natural to proceed on the assumption that neurons by themselves are sophisticated processors consisting of large amounts of internal memory and built-in programs for using that memory to solve problems with which their environments confront them.

We are not, of course, suggesting that a neuron is literally built like a computer processor but only that its level of processing power, implemented by means of ionic processes, biochemical reaction systems, and other mechanisms, may be more fruitfully compared with that of a microprocessor than with that of a logic gate. Since a microprocessor need not be programmed to perform very complex computations, nothing precise is implied by this metaphor. It simply raises questions about the level of functional complexity at which one might place familiar biological "primitives" such as neurons.

Some early attempts to produce machine intelligence were based on the hope that networks of neuron-like components could "self-organize" from initially unstructured configurations. Some reasons for the failure of this

approach to yield really interesting behavior were chronicled (notably by Minsky and Selfridge, 1960, and Minsky and Papert, 1969), and the mainstream of interest largely shifted to the more symbolic approach that characterizes most current artificial intelligence research. While early network studies seemed to be concerned with the components out of which systems were constructed, the symbolic approach was not. Nilsson (1974), for example, remarked that:

. . . knowledge about the structure and function of the neuron - or any other basic component of the brain - is irrelevant to the kind of understanding of intelligence that we are seeking. So long as these components can perform some very simple logical operations, then it doesn't really matter whether they are neurons, relays, vacuum-tubes, transistors, or whatever.

Klopf (1979, 1981) remarks on how the point of view expressed by this statement downgrades the neuron (especially in light of the extraordinary complexity of neurons that is being revealed by neuroscience research) and, in particular, tacitly denies the possibility that neurons might be performing more than simple logical operations. In what we are calling the distributed processing metaphor, the structure of the system as a network of components is important, as are the functional properties of the component processors. The emphasis is not on how very simple components can interact to solve some problems but rather on how components that are already

capable of solving problems can interact to solve more complex problems. It is the study of networks at a much higher level of organization. Logic gates and transistors, or other lower level physical components, may still be unimportant (as long as the processors operate), but the system's structure as a network of components is a central concern. And self-organization is again a central concern. Control strategies are sought that permit meaningful cooperation among the processors for problem solving. A good example of the distributed processing approach to problem solving is provided by Lesser and Erman (1979).

We therefore arrive at the view of a neural network as a very large distributed processing system in which each neuron, or microprocessor, is capable of solving certain nontrivial problems by itself. The resulting metaphor is vastly different from that which early network theorists brought to their research. We have little doubt that this metaphor will be replaced by others before we understand animal brains, but it can form the basis of a very large step from our current state of knowledge. But what sort of problems might this neural microprocessor be capable of solving?

In a 1972 monograph, A. H. Klopf described a theory of memory, learning, and intelligence based on a view of

neural processing that emphasized the fact that neurons are living organisms that have survived a long evolutionary process (Klopf, 1972, 1979, 1981). He suggested that progress in understanding natural intelligence might be achieved by a study of goal-seeking systems of goal-seeking components. Instead of viewing any form of goal-seeking behavior as an emergent property of a system composed of non-goal-seeking components, Klopf suggested that sophisticated goal directed behavior arises from interacting components that are themselves goal-seeking and act according to their self-interests. Adaptive capabilities are thus pushed down the structural hierarchy to basic levels. The biological correlate of this view is Klopf's hypothesis that neurons are the goal-seeking components of animal brains and that they possess behavioral strategies permitting them to make progress toward their goals.

In proposing what a neuron's goal might be, Klopf takes issue with the widely held view that homeostasis is the goal of living organisms. Homeostasis is the condition in which all critical variables are maintained within acceptable ranges. Ashby (1960) identifies adaptive behavior with behavior leading toward and maintaining homeostasis and described a device called a homeostat that could maintain homeostasis in the face of a wide range of environmental perturbations. Klopf proposed that the goal of an organism

is not homeostatic equilibrium but rather a condition in which specific internal variables are in a maximal condition. He proposed the term heterostasis to describe the condition in which a given variable is maintained at its maximal level.

Klopf makes a subtle but important point. Given general agreement on the processes underlying the evolutionary process, living organisms will possess behaviors or strategies for maintaining environmental conditions favorable for survival and reproduction. But this does not imply that the goal-seeking strategies used by an organism work directly toward maintaining conditions favorable for survival and reproduction. Animals do possess equilibrium-seeking homeostatic mechanisms, but the goal of an animal may be quite different from homeostasis or even survival. A direct goal of an animal may be to maximize the occurrence of certain types of sensations. Which sensations play this role in an animal's life is determined by the differential survival of past generations. Depending on the particular environment, the maximization of the occurrence of certain sensations usually does lead to survival and reproduction. But the direct goal of the animal need not be survival and reproduction. Survival and reproduction are side effects (but certainly not by accident!) of an organism's goal-seeking strategies.

Klopf hypothesized that neurons are goal-seeking organisms that use a particular strategy for maximizing the occurrence of particular types of neural "sensations" and for minimizing the occurrence of others. The strategy takes the form of a rule for altering variable synaptic transmittances:

After a neuron fires, it waits for a few hundred milliseconds or more to see how it will be affected by the action it has taken. If it experiences further depolarization within a second or so, it increases the effectiveness of the excitatory synapses that led to its firing in the first place, thereby increasing the probability that it will fire the next time some fraction of these synapses is active. If, however, the action of firing is followed within a second or so with the experience of hyperpolarization, the neuron then increases the effectiveness of those inhibitory synapses that were active when it fired. In this way, the probability of responding again to the input configuration has been diminished. Thus, the neuron views excitation as reward and inhibition as punishment. (Klopf, 1981)

Klopf is suggesting here that a single neuron implements the Law of Effect, or can be conditioned in an operant or instrumental manner where depolarizing and hyperpolarizing events act, respectively, like appetitive and aversive stimuli. Klopf called a component operating according to these principles a heterostat.

Our research has been directed toward determining whether or not Klopf's theory can provide the basis for progress in understanding adaptation, learning, and

intelligence in general. Has this type of component already been investigated? Can devices resembling heterostats interact to form higher kinds of goal-seeking behavior? This report describes the conclusions we have reached and contains some of the computational results we have produced. Briefly, we have found that the class of learning rules suggested by Klopf has not been extensively investigated. It is generally believed that the important features of the Law of Effect were incorporated into various learning rules that were investigated with little dramatic success, but our research has led us to question this belief for a number of reasons that we explain in this report (especially Section 2). We think the potential for real progress using the approach proposed by Klopf is very great.

Our research strategy has been one in which the logical and mathematical properties of adaptive systems have been emphasized. Although we have tried to make as much contact with neurobiological data as is possible, we have not attempted to model specific neural processes. The adaptive components we have investigated have many neuron-like attributes and were motivated by Klopf's neural hypothesis, but we purposefully refer to them as "adaptive elements" rather than neural models. Although an understanding of electrical and chemical processing of neurons is increasing rapidly, we think it is premature to propose an extensive

and detailed neural model. We argue that computations of the complexity required are clearly possible at a neural or synaptic level and that mechanisms of the required character exist, but we do not rely on this line of empirical support. We have instead concentrated on carefully specifying a range of problems, and on designing mechanisms capable of solving them. We believe that these problems are likely to be involved in any attempt to produce systems that can adapt and learn, including highly symbolic systems. We cannot say that any such system must solve these problems using our methods, but we do think that our methods are readily applicable and can form useful parts of systems that are more complex than those illustrated in this report. Our illustrations were chosen for their simplicity and clarity, and should be interpreted as simple examples of classes of problems that can occur in a variety of different contexts.

The report is written so that an understanding of most sections does not require the reading of preceding sections. However, the sections are very closely related. Issues are raised in some that are more adequately addressed in others. A brief summary of each section is provided here.

Section 2 contains an extensive, and sometimes technical, analysis of adaptive system theory in which we attempt to place various adaptation or learning methods,

including Klopf's proposal, into a general framework. Our analysis is not exhaustive, nor do we claim that it is definitive. It represents our attempt to understand a number of issues that arose in the course of our research. Attention is not restricted to models of learning processes cast in biological or neural form. A large body of highly developed theory exists in the field of control engineering, and we try to include problems and methods from this field into our general framework. We make what we think is a very important distinction between two types of search: those in which you can recognize what you are looking for when you find it, and those in which this is not possible. We argue that although the perceptron learning rule (Rosenblatt, 1962) is often thought to capture the important features of closed-loop reinforcement learning, it does not. Even aside from the usual objections to the perceptron based on its linearity, it is capable only of a very restricted type of learning. We delineate a class of problems requiring very general learning capabilities that has received very little attention in the past. Systems like Klopf's proposed heterostat are capable of solving problems of this kind and have not been extensively investigated.

Section 3 is a roughly chronologically ordered description of a number of learning rules we have investigated. Rather than being a defense of all of these

efforts, it is intended to help others who may become interested in this approach; perhaps they can benefit from our experiences.

Section 4 is a discussion of a learning rule that incorporates some of the features of Klopf's proposal but not all of them. At a particular stage in our research, we felt that we had reached an understanding of certain issues and could contribute to both adaptive network modeling and animal learning theory. The learning rule described includes some of the temporal dependencies suggested by Klopf but does not implement a closed-loop rule whereby the consequences of actions cause appropriate changes to take place. We discuss the advantages of a form of reinforcement different from that originally suggested by Klopf and relate its consequences to data about animal learning in classical conditioning experiments. We also discuss the learning rule in light of physiological and biochemical data.

In Sections 5 and 6 we present computer simulation results that illustrate the behavior of simple networks of adaptive elements having some of the properties of Klopf's heterostat. Section 5 contains a description of a problem that we have called the associative search problem and a network, called the associative search network, that can solve the problem under certain conditions. Section 6

illustrates the behavior of an associative search network that controls locomotory behavior in a simple spatial environment. We chose this example in order to provide a graphical way of demonstrating the kind of problem an associative search network can solve. We did not try to model the locomotory behavior of any particular animal, nor did we try to show all of the system's capabilities. It is the simplest example we could invent.

Section 7 describes a system that illustrates some of the more interesting consequences of learning rules that take careful account of the temporal factors involved in learning. The system is able to construct a predictive model of its environment and then use it to evaluate the consequences of proposed, but not overtly taken, actions. We illustrate its behavior in a simple latent learning task. Again, our intent was to illustrate in as simple an example as could be constructed how networks can be synthesized to perform this type of processing.

Section 8 is a discussion of goal-seeking systems composed of goal-seeking components. We discuss in rather speculative form how higher-level goal seeking behavior can be expected to arise from collections of components that always act according to their own interests. We discuss what capabilities the components should have in order for

them to cooperate. The relevance of certain concepts from the theory of games is discussed. We then discuss the plausibility of Klopf's hypothesis from a biological rather than from a theoretical point of view. Goal-seeking strategies known to exist in unicellular organisms are discussed.

Section 9 provides a brief account of the major observations made in the course of this study.

The appendices contain information that either supplements the main text or is somewhat peripheral. Appendices A and B respectively contain a mathematical analysis and a formal description of the model discussed in Section 4. Appendix C presents the initial stages of our efforts to develop adaptive strategies to alter some of the parameters of the learning rules in order to accelerate convergence. The results have not yet been integrated with the major line of our research. Appendices D and E contain detailed descriptions of the environment and adaptive network, respectively, discussed in Section 7. Finally, appendices F and G document some of the software research tools that we have developed. The systems described are interactive color graphics programs that allow the design, display, and simulation of networks and spatial environments.

SECTION 2

ADAPTIVE SYSTEM THEORY

2.1 Introduction

It is notoriously difficult to precisely characterize those problems whose solutions can be said to require adaptation or learning. One major criterion for calling a problem or mechanism adaptive seems to be that it involve some form of goal-seeking. The goal may be the maintenance of critical variables within prescribed limits as suggested by Ashby's characterization of adaptation as homeostasis (Ashby, 1960), or it may be to show continued improvement in performance according to some measure on the basis of past individual experience as suggested by Holland's characterization of an important aspect of adaptation as function optimization (Holland, 1975) and Klopff's theory of heterostasis (Klopff, 1972, 1979, 1981). Any problem solving method might be said to have as a goal the problem's solution, but the terms adaptive and learning seem to be reserved for those methods that do not have built-in

knowledge about exactly what the goal is or how to obtain it.

Rather than risk making precise definitions that are likely to be revealed as inadequate upon more careful thought, we focus on a number of specific examples of problems and methods that are often thought to have something to do with adaptation or learning. We present a number of distinctions that we have found useful in guiding our thinking about these problems and methods as we attempted to compare and contrast them. Our perspective is general enough to include most of the problems typically regarded as requiring some degree of adaptation or learning for their solution, but we do not wish to claim that all of the problems we discuss are correctly so characterized. On the contrary, we wish to provide a perspective general enough to include significant adaptation and learning problems as well as the most simple ones in order to illustrate how restrictive many of the problems really are. Despite our attempt to be as precise as possible, we do not wish to give the impression that these distinctions and the problem classification they imply are complete and definitive. Our intent is to explicate the current state of our own understanding.

Of the very many useful distinctions that can be made

between various adaptation and learning problems and mechanisms, we have chosen to focus only on a few that we think have particularly great significance because they help elucidate some of the folklore about adaptation and learning that we think is misleading. One of our goals is, in fact, to argue that both the view of adaptation as homeostasis and the view of adaptation and learning as primarily function optimization are inadequate in providing a perspective complex enough to usefully elucidate even the simplest adaptive or learning behavior exhibited by animals. Another of our goals is to explain what we think is the fundamental novelty of Klopff's theory of neural plasticity. As a result of this selectivity, many important aspects of adaptation and learning will not be adequately addressed, and we will not attempt to discuss the enormous variety of special methods and subclasses delineated in the adaptive systems literature. It is also not our intent to provide a thorough survey or an extensive bibliographical source. We suggest that our remarks be interpreted as an initial attempt to focus on a number of issues that seem to be frequently misunderstood. In Section 8 is a closely related discussion in which some of the issues we raise here are related to the potential of different types of goal-seeking systems as components of larger systems.

2.2 Problems Versus Mechanisms

Our discussion is divided into two major parts. The first focuses on various basic adaptation and learning problems rather than on methods, algorithms, mechanisms, or strategies. By an adaptation or learning problem we mean a task with respect to an environment whose accomplishment would be regarded as demonstrating adaptation or learning. We therefore discuss problems by considering the nature of the interaction between a system and its environment. Problems are characterized by the kinds of signals available to the system for influencing its environment and for sensing its environment's condition, the amount and nature of a priori knowledge available to the system or to its designer about the environment, and various characteristics of what constitutes a solution to the problem. The second part of our discussion deals with particular examples of these basic problems as well as methods, mechanisms, or strategies that are commonly used to solve them. Table 2.1 summarizes our view of these problems and methods in light of the distinctions we make. Periodic reference to this table may help the reader retain an overview of our presentation.

We have found the distinction between problems and methods useful since it is possible for a given problem to

TABLE 2.1
A CLASSIFICATION OF PROBLEMS AND METHODS

	Nature of Control Over Input Open-loop/Closed-loop	Number of Control Situations Single/Multiple	Knowledge of Control Surface Complete/Partial	Knowledge of preferences Error-correcting/Extremum search
Pattern Recognition	X	X	X	X
Associative Memory	X	X	X	X
Classical Conditioning	X	X	X	X
Prediction	X	X	X	X
Perceptron	X	X	X	X
Function Optimization	X	X	X	X
Closed-loop Control	X	X	X	X
Kineto-kinesis	X	X	X	X
Tropo-toxis	X	X	X	X
Error-correcting adaptive control*	X	X	X	X
Homeostat	X	X	X	X
Extremizing adaptive control*	X	X	X	X
Error-correcting learning control*	X	X	X	X
Adaptation accumulating homeostat	X	X	X	X
Reinforcement learning control*	X	X	X	X
BONES	X	X	X	X
Instrumental conditioning	X	X	X	X
Heterostat	X	X	X	X

* refers to second-level problem

be solved by several different methods, and it is possible to apply a given method to several different problems. An understanding of a method therefore requires knowing what problems it is capable of solving. In addition, most adaptive systems consist of a hierarchy of components each faced with its own basic problem. By first considering a number of basic adaptation and learning problems, we can discuss various familiar methods by describing the basic problems their parts are designed to solve.

We leave open the question of what determines the distinction between a system and its environment. Our view is simply that any part of any system can be viewed as a system interacting with an environment. Since a problem is determined by the nature of the system/environment interaction, redrawing the system/environment interface line merely implies that what is then called the adaptive system is faced with a different type of problem or the same type of problem at a different level.

Perhaps the most important consequence of the distinction between problems and methods is that it permits us to consider separately two different types of limitations on the capabilities of methods, mechanisms, or strategies. On the one hand, a method may be limited in its ability to solve problems of a given type that are more complex than a

certain level. On the other hand, a method may also be limited in its ability by being able to solve only a certain type of problem. For example, the usual objections to an adaptive system based on perceptrons rest on the fact that they can only perform linear discriminations. A very different type of objection is that perceptrons, even if they could implement arbitrarily complex discriminant functions, are limited because they require an environment capable of providing them with a specific type of detailed information. In this report we focus almost exclusively on limitations of this latter type since they are not commonly appreciated, and their discussion leads, we think, to constructive and novel suggestions about how to solve difficult and general adaptation and learning problems.

2.3 Basic Adaptation and Learning Problems

Figure 2.1 shows an adaptive system AS interacting with its environment E. At each moment t of time the adaptive system receives a stimulus signal $S(t)$ from its environment and sends an action signal $A(t)$ to its environment. By signal we do not mean a unidimensional form of stimulation. The signal S may be a very complex multidimensional sensation corresponding to some entire environmental situation. Similarly, A may be a very complex action. We have used broad arrows in Figure 2.1 to suggest a possible

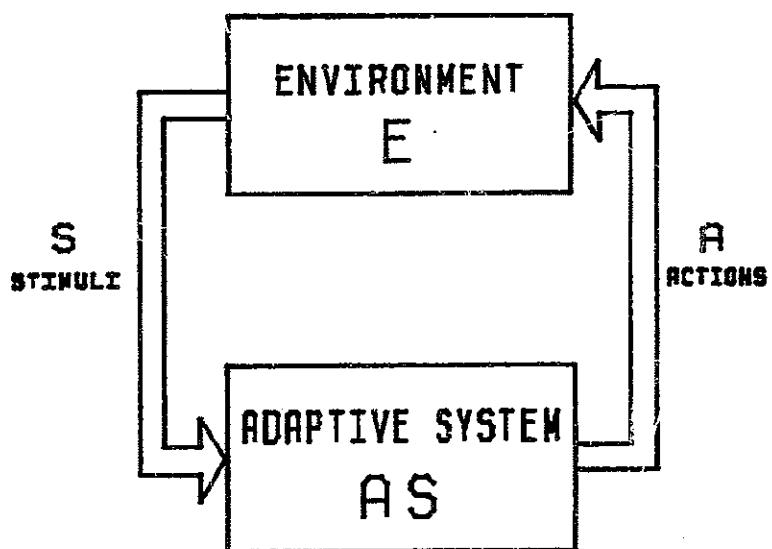


FIGURE 2.1. An adaptive system interacting with its environment. The adaptive system receives stimuli signals S from the environment and sends action signals A to it. The broad arrows indicate a possible flood of information flowing between the adaptive system and its environment.

flood of information flowing between the system and its environment. Adaptation and learning problems differ greatly as to how much of this information can be used.

All of the problems we will discuss can be viewed as special kinds of control problems if we accept the very general description of a control problem given by Aizerman et al.(1964) and quoted by Mendel (1970): "...the very problem of automatic control as a whole can be considered to be the problem of assigning the input situation to one or another class, and to generate the optimal response as a function of that class." In our terms, the input situations correspond to the possible values of the stimulus signal S and the possible responses to the values of the action signal A. We will find it convenient to refer to the concept of a control situation as defined by Mendel and McLaren (1970). For our purposes a control situation is a subset of values of the stimulus signal S for which a single action, or "control choice", is optimal. In other words, a control situation is a collection of sensory situations in each of which the same action is best. The mapping that associates to each possible situation the optimal action for that situation is variously known to control theorists as a switching function, switching surface, or control surface (Mendel, 1970).

This view is adequate for some of the observations we wish to make, but a slightly modified view seems to be more general. This view requires a shift in perspective from the output of the system to its input (cf. Powers, 1973). Let the adaptive system AS possess a preference ordering on its possible stimulus signals. In the most general case there may be a preference ordering only on sequences, or time trajectories, of stimuli, but here we let the preference ordering apply simply to instantaneous stimulus patterns. This simplification prevents us from addressing some very important issues, such as short-term versus long-term goals, but is sufficient for most of the points we wish to emphasize here. In terms of a preference ordering on stimulus patterns, the problems we consider are all special cases of the general problem of acting in response to each stimulus pattern so as to cause stimulus patterns to occur that are as preferable as possible according to this preference order. We call the most preferable inputs possible in any situation the optimal inputs, and we say that a response is optimal if it causes an optimal input to occur. The control surface assigns each stimulus pattern to an action that is an optimal response to that stimulus.

Often a system's preference order is determined by the magnitude of a real valued measure variously known as a criterion function, payoff function, index of performance,

or utility function. Larger magnitudes of this measure are preferable over smaller ones. In some cases, reversed measures are used, such as a measure of disutility, in which case smaller values are preferable to larger ones. In either case the resulting order is total: any value is either greater than, equal to, or less than any other value. Other problems are characterized by a partial preference order: there are some inputs that are neither better than, worse than, nor equal to some others. These problems often arise if there is not a single measure but a set of distinct scalar measures each characterizing a different aspect of an input. In these cases it may not be clear what optimum means since tradeoffs among the individual measures may occur. If a useful scalar measure can be assigned to each vector (as in linear programming), then the problem is reduced to the totally ordered case discussed above. If this is not possible, or has no reasonable interpretation, then the definition of optimality varies depending on the application. These are multicriterion decision problems. Ho (1970) has suggested the term generalized control theory for control problems with multiple criterion functions. It includes vector valued optimization problems, game theory, and involves such notions as pareto-optimality, Nash points, cooperation, side payments, coalitions, and even trust and threat. These are the kinds of problems that arise when collections of interacting goal-seeking systems are studied,

and the theory is extremely relevant to the research we have begun. In Section 8 we extensively discuss the relevance of multicriterion decision theory to adaptive systems research and to theories of mental function. However, for our present purposes we restrict attention to problems characterized by total preference orderings.

2.3.1 Some Basic Distinctions

The adaptation and learning problems we discuss differ in how the adaptive system can influence the control situations it faces, how many different control situations there are, how much about the control surface is known from the start, and knowledge about the preference ordering. We introduce a number of important distinctions based on these features of a problem. The most obvious examples of each type of problem will be cited in order to help make the distinctions clear.

2.3.1.1 Nature of Control Over Input - Referring to Figure 2.1, a problem is open-loop if its solution does not require AS to influence its input. In other words, the problem could still be solved even if it were impossible for AS to exert any influence over its input. Note that we are not defining an open-loop problem to be one in which AS cannot

influence its input, but rather one in which any such control is irrelevant. A problem is closed-loop if its solution does depend on the ability of AS to influence, however indirectly, the input signals it receives from E. A problem is closed-loop if its solution becomes impossible in the absence of such control. Examples of closed-loop problems are function optimization and feedback control problems.

According to the general view we presented of an adaptation or learning problem as one in which a system is controlling its input, it would seem that, by this definition, no such problem can be open-loop. This is indeed true, but there is a general class of problems usually associated with adaptation and learning that turn out to be open-loop from a certain point of view. These are the usual forms of pattern recognition problems, including those solved by perceptrons and related learning rules. Although these types of problems are very important, one of the goals of our discussion is to distinguish them from a very different class of genuine closed-loop problems.

It is useful to distinguish between two types of closed-loop problems depending on the type of control AS has over its input signals or its control situations. In control theoretic terms, this is a distinction between

different kinds of environment controllability. In some cases the adaptive system can exert direct control over its input signals. By this we mean that, from the point of view of AS, E merely realizes a mapping from AS actions to AS inputs. Without significant loss of generality, we can assume that there is a fixed delay through the environment. Then $S(t) = f(A(t-1))$ where f is a mapping from AS actions to AS inputs. This implies that the response of the environment to any given action will always be the same. Another way of stating this is that from the point of view of AS, E is a system without memory. Function optimization problems are examples of problems in which this kind of direct environmental control is assumed.

Other closed-loop problems are characterized by the fact that AS has only modulatory control over its inputs. These are problems in which E appears to AS to have memory. Consequently, a given action by AS may produce different environmental responses depending on the internal state of the environment. Without undue loss of generality, we can say that $S(t) = f(A(t-1), Q(t-1))$ where $Q(t-1)$ is the internal state of E at time $t-1$. Most control problems are characterized by this property. For example, a thermostat cannot directly force the room temperature to a given value but can only modulate it.

Most problems faced by biological systems also have this property. One interesting class of such problems consists of those arising from movement in space. If the input to AS corresponds to an indication of its spatial position, and an action corresponds to a movement, then the spatial environment E can only be modulated. A given movement does not take the animal to the same place each time it is performed (unless it is a high level go-to-place-X movement). The destination depends on the initial location and orientation. A location and orientation in space is an internal state of a spatial environment.

2.3.1.2 Number of Control Situations - A problem may have a single control situation, an infinite number, or any number in between. The most useful distinction that can be drawn here is between problems having one control situation and those having multiple control situations. The function optimization problem is an example of a one control situation problem. The task is to find the minimum or maximum of a (usually real valued) function. A mechanism designed to solve a function optimization problem may include parts that solve multiple control situation problems (e.g. perform one action if the payoff value has decreased, and another one if it has increased). But with respect to the environment that evaluates the payoff function for each

action of the adaptive system, the problem has one control situation since there is only one function to optimize. The performance of a given action always results in the same function value. Pattern recognition problems are examples of multiple control situation problems. Each control situation corresponds to a pattern, and the task is to assign each input signal to the correct control situation.

2.3.1.3 Control Surface Knowledge - A problem can be characterized by the amount of knowledge available from the start about the control surface. It is useful to distinguish those problems in which there is complete knowledge from those in which there is only partial knowledge. The problem solved by a simple servomechanism, such as a thermostat, is an example of a problem in which the control surface is completely known and fixed from the start. This is a multiple control situation problem (one control situation occurs when the room temperature is too high, the other when it is too low), and the correct action for each situation is built into the thermostat: furnace off in situation 1, furnace on in situation 2. This specification of the control surface is based on a priori knowledge about the nature of the environment. Here it is knowledge about what a furnace does to room temperature. The partial knowledge counterpart of this control problem is known as an adaptive control problem. These problems occur

when the dynamics of the environment are not known with enough detail to permit the prespecification of the control surface. In these cases, the control surface must be determined from system experience

Any mechanism designed to solve a problem with partial control surface knowledge must contain, at some level, a component that solves a problem by using a completely prespecified control surface. If this were not the case, it would not be possible to consider designing a mechanism in the first place. For example, at its lowermost level a device might be trying to solve a problem with partial control surface knowledge. A higher level component of this device therefore faces the problem of synthesizing this control surface and may do so according to its own prespecified control surface.

2.3.1.4 Knowledge of Preferences - This is a very subtle issue but one that is fundamental. In fact, we think this issue deals with a major tenet of Klopff's theory which includes the claim that homeostasis, or equilibrium, is very far from being the complete story when considering the adaptive behavior of animals. For simplicity in the following discussion we assume there is a single optimal input, but our remarks should be understood to apply to the more general case in which there are different optimal

inputs depending on the environmental state. Some problems have the property that the optimal input is already known from the start. In these problems, the system "knows" exactly what it wants in the sense that it can recognize it when it finds it, but must manipulate its environment or its own response system in order to make the desired input appear. The problem is in this manipulation rather than in the characterization of the optimal input. In other problems, all that is known about the optimal input is that it maximizes the preference ordering. Nothing is known about the optimal input that permits the system to perform it or recognize it "on sight".

These two types of problems are vastly different but both involve the idea of search. Some examples may clarify what we mean. Consider the problem of searching for a specific item in a list, a particular name in a phone directory, for example. The name is known from the start, but the control over the phone directory is not direct. We cannot approach the directory and command it to open immediately to the appropriate page. A similar problem is solved by a thermostat. The desired temperature is determined by the setting, and the problem is to cause the thermometer to register that temperature. This is analogous to the task of causing the phone directory to present the desired name to the searcher.

These problems should be contrasted with that of searching for, let us say, the best, most expressive adjective to use in a specific phrase. If we know the best adjective from the start, we can simply use it since we have complete control over what we write. The problem is to find the adjective that is best. All we know about it from the start is that it is better than all the other adjectives. This is a much different problem and would remain so even if there were a well-defined, easy to determine preference measure for adjectives, and even if there existed an exhaustive alphabetically ordered listing of adjectives. The crucial difference is the following: the search of the telephone directory can stop when the desired name is found, but the search for the best adjective ought, ideally, to continue until every possible adjective has been evaluated. In the former case, optimality is a local property of individual trials whereas in the latter case it is a property of the entire set of possible trials.

If the optimal input is known from the start, it is often possible to define a measure of error in order to guide the search. The error measure may have just two values, signaling error or no error, as in the search of an unordered list, or it may have many values which indicate the magnitude and perhaps also the direction of error as in the search of an alphabetized phone directory and the

temperature control problem. We therefore call problems in which the optimal input is known from the start error-correction problems. We emphasize that the desired input must be known a priori in order to define the error. We call problems in which the optimal input is not known from the start extremum search problems.

The profound difference between these types of problems is obscured by the fact that error-correction problems are special kinds of extremum search problems. If errors can be ordered so that it makes sense to say that one error is larger than another, then an error-correction problem can be solved by minimizing the error which is an extremum search problem. But extremum search problems that result from error-correction problems all have a very restricted form. The error function is either known to be unimodal (that is, to have a single local minimum) or, if there are several local minima, it is known that each locally minimum value is equal to the global minimum (e.g., searching for any of several items in a list). If the error function does not have these properties, the error measure is not a good one.

These restricted types of extremum search problems can be solved very easily using a method that can be viewed in two ways depending on whether one emphasizes the error-correction or extremum search aspect of a problem.

The method appears as an equilibrium-seeking negative feedback method if the error-correction view is emphasized. Since the optimal input is known from the start, it is possible to define a control surface that effectively sets up a "restoring force" around the optimal input by means of negative or "deviation counteracting" feedback. The method consists of designing a servomechanism whose set-point is the desired input. Negative feedback causes the resulting servomechanism/environment composite system to have as its equilibrium state that which produces the optimal input to the servomechanism. When the extremum search aspect of an error-correction problem is emphasized, on the other hand, this same method appears as a simple type of gradient descent procedure designed to minimize the squared error. Like gravity, the restoring force appears as the force causing the state to descend toward, and then remain at, the nearest local minimum. However one views this method, it is clear that it can successfully solve only a very restricted class of extremum search problems. To understand these restrictions most convincingly, imagine trying to use a thermostat in an environment that determines the temperature, and hence the error signal, by some unusual function of the thermostat's action. The thermostat can minimize only a very small class of such error functions!

To summarize, error-correction and extremum search

problems differ in a number of ways.

- 1) An error-correction problem is solved when the desired input is attained even if all of the possibilities have not been evaluated. An extremum search problem, on the other hand, is completely solved only when the entire range of possibilities has been explored. Obviously, a search space can be so large that the extremum search problem can never be solved in practice. In such cases it becomes important for the adaptive system to show continuing improvement or sufficiently high cumulative performance. These requirements can be thought of as other adaption or learning problems requiring preference orderings on sequences of inputs.
- 2) Any error-correction problem can be transformed into an extremum search problem but not vice versa.
- 3) An error-correction problem can be solved by an equilibrium-seeking mechanism. If the adaptive system/environment system attains this equilibrium, or is set to it, then the response of the adaptive system will not change. On the other hand, it is often useful for an extremum search mechanism to be incessantly active to improve its chances of solving the extremum search problem or for showing continuing improvement.
- 4) The hard part about an error-correction problem is not to

determine what the optimal input is but rather to cause it to occur. The hard part about an extremum search problem, on the other hand, is to determine what the optimal response or state is. The general adaptation problem, and one which seems to be commonly faced, may be to desire some optimal situation and know neither what it is nor how to attain it!

2.4 Particular Problems and Methods

2.4.1 Pattern Recognition Problems

From our general perspective, a number of different problems appear as the same basic open-loop problem which we call the pattern recognition problem after its most familiar example. Other examples are associative memory storage and retrieval, certain forms of system identification, and the problem an animal confronts in a classical conditioning experiment. By a pattern recognition problem we mean a problem that is similar in logical structure to the traditional formalization of pattern recognition (e.g., Duda and Hart, 1973). Pattern recognition in a broader sense is really a complex collection of problems. In the traditional formalization, pattern recognition is treated as a relatively passive process that does not involve the use of information extracted from patterns for explicit control purposes. All of the examples of what we are calling the

pattern recognition problem can be related to the interaction shown in Figure 2.1 by assuming that every signal S from the environment consists of two parts: $S = (X, Z)$ where both X and Z can be multidimensional signals. It is convenient also to assume that Z can take on a distinguished value called 'null' in addition to other possible values.

PATTERN RECOGNITION - The pattern recognition problem appears as follows: X_a , $a=1, \dots, k$, is a "training set" of patterns having respective classifications Z_a , $a=1, \dots, k$. For example, if each X_a were some representation of a hand written alphabetic character, then each Z_a could be the name of the correct recognition class label 'A', 'B', etc. After a number of presentations to the system by the environment of the pairings (X_a, Z_a) , $a=1, \dots, k$, the system should respond with Z_a to each input $(X_a, 'null')$, $a=1, \dots, k$. That is, it should "learn" to correctly classify the patterns in the training set. Additionally, it should be capable of generalizing from its training experience and correctly classify patterns not included in the training set. Except for these very important generalization capabilities, the form of learning accomplished by pattern recognition systems is the same as that performed by a standard computer memory: for input (X_a, Z_a) , Z_a is a data item to be stored at address X_a so that during a read cycle, address X_a will produce Z_a as output. The real problem in pattern recognition is, in

fact, not really the problem we have just described but rather the feature selection problem. In order for meaningful generalizations to be made, patterns must be represented by appropriate signals. For our present purposes we view the problem of finding appropriate representations as a type of problem different from the formal pattern recognition problem. What is generally known as scene analysis is, for example, largely concerned with the extraction of useful features from visual patterns. Hanson and Riseman (1978), provide a good view of current scene analysis techniques. Duda and Hart (1973) and Mendel and Fu (1970) provide good overviews of pattern recognition methods.

Formulated in this manner, the pattern recognition problem and related problems discussed below are open-loop. They do not require the adaptive system to exert any control over its input. However, it is common to view these problems in an equivalent manner that does involve a closed loop. In this formulation the environment presents the system with each pattern X_a , to which the system responds with some estimate Y_a of the correct classification label for X_a . Then the environment presents the system with the error between the correct classification Z_a and the system's estimate Y_a . This error is usually $Y_a - Z_a$, that is, it is a signed error. Viewed in this way the pattern recognition

problem fits into the general framework we are using. The most preferable error is zero. We shall argue below, however, that this closed-loop view does not really alter the fundamental open-loop character of pattern recognition problems. This is an important claim since the perceptron, on which much early attention was focused, solves this type of problem. We discuss the perceptron learning rule in some detail below.

ASSOCIATIVE MEMORY - There are many different types of associative memory structures. Some, such as those discussed by Foster (1976), are currently feasible alternatives to the familiar form of computer memory. Others, such as those discussed by Amari (1977), Anderson et al. (1977), and Kohonen (1977), are analog in character and have been proposed as models of biological memory. These latter associative memory structures can be viewed as solving pattern recognition problems. In most pattern recognition tasks as discussed above, each signal Z is a unidimensional label for a recognition class. For associative memories, the signals Z are multidimensional patterns. In these cases, each X_a is a "key" and the corresponding Z_a is a pattern to be associated with X_a . The training phase is interpreted as the period in which the key-pattern associations are stored in the associative memory for later retrieval. The retrieval process can be interpreted as the classification of the input key. The

ability of pattern recognition systems to generalize from their training set appears as the error tolerance and associative generalization capabilities of these memory structures. In Section 4 we provide a further discussion of these types of associative memory structures.

SYSTEM IDENTIFICATION - As noted by Kohonen (1977), the associative memory task can be viewed as a system identification task if it is assumed that there is a functional relationship, unknown to the associative memory system, between keys and patterns. Figure 2.2 shows an associative memory system receiving information from an unknown system. Each key (X_a) sent to the associative memory is an input signal to the unknown system, and the corresponding pattern to be associated (Z_a) is the unknown system's output. Successfully storing all of these associations can be viewed as having successfully identified the unknown system's input/output function. This basic view can be elaborated in many ways to handle cases in which the unknown system has internal memory. See, for example, Tsypkin (1971).

CLASSICAL CONDITIONING - A classical conditioning experiment can be viewed within the pattern recognition framework by letting patterns X_1 and X_2 correspond respectively to CS presence and CS absence, and by letting Z_1 and Z_2 correspond

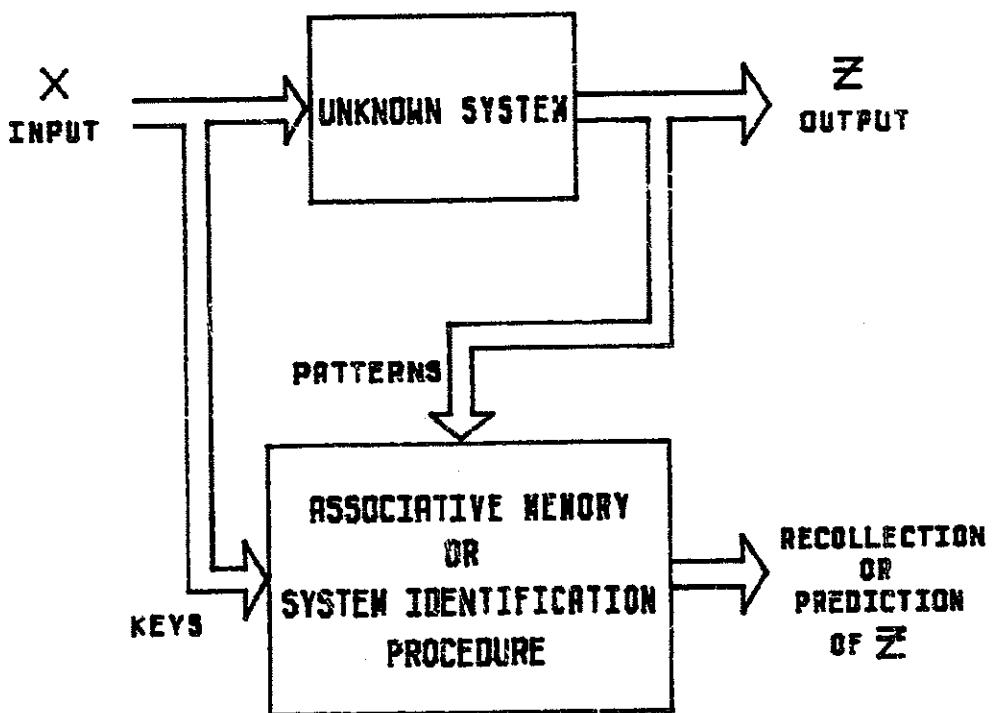


FIGURE 2.2. The associative memory problem viewed as a system identification task. There is an unknown functional relationship, implemented by a system with an unknown input-output function, between keys and patterns.

respectively to UCS-UCR occurrence and UCS-UCR omission. Trials consist of repeated presentations of the pairs (X_a, Z_a) , $a=1,2$. After sufficient training, $(X_1, 'null')$ elicits Z_1 (the CR) and $(X_2, 'null')$ elicits Z_2 , the absence of the CR. This view captures some of the logic of classical conditioning. It illustrates its open-loop nature and its relationship to pattern classification. However, as we point out in Section 4, it is not an adequate account of classical conditioning even though it seems to be generally regarded as such by neural network theorists. It neglects the fact that the distinction between the CS and the UCS is essentially temporal and that the CR anticipates the UCR.

PREDICTION - It is more appropriate to model the behavior elicited in classical conditioning as a variant of pattern recognition or system identification known as prediction. Here, the signals S from the environment are not required to have two distinguishable parts. The simplest case of the prediction problem is solved by the adaptive system when $A(t) = S(t+1)$ for all t . In other words, the action of the adaptive system anticipates its input signal (here by 1 discrete time step). This is the same as the pattern recognition problem discussed above if the two part signal (X, Z) is taken to be $(S(t-1), S(t))$. We discuss classical conditioning and the prediction problem extensively in Section 4.

According to the distinctions discussed above, all of the problems we have called pattern recognition problems are characterized as follows:

- 1) The adaptive system AS in each case has no control over its input. Consequently, these are all open-loop problems. However, they can all be transformed into equivalent closed-loop problems by letting the environment provide only patterns and error signals. It must be noted that some types of system identification procedures generate test signals as input to the unknown system. This makes the identification problem closed-loop in an essential way. We think that it is not misleading, however, to regard the open-loop identification problem discussed above as the core system identification problem.
- 2) These are multiple control situation problems. The pattern classes of the pattern recognition problem are its control situations. For the associative memory task, a set of keys, each of which should elicit the same output pattern, is a control situation. For the prediction problem, a control situation consists of input signals that are all followed by the same input signal. Some of these problems obviously have an infinite number of control situations.
- 3) There is partial knowledge of the control surface. In the pattern recognition task, for example, the control

surface is the mapping that assigns the correct classification to each input pattern. If it is known initially, then no problem needs to be solved.

4) These problems are all error-correction problems. The problem is not to find the best response Z_a for each pattern X_a but rather to cause Z_a to occur as a response to X_a . The best response is directly provided to the adaptive system by the environment (assuming noiseless conditions).

2.4.2 Clustering

The problem of clustering is closely related to the pattern recognition problem. The basic clustering problem is to classify input patterns not according to the instructions of an external teacher, but according to their similarities and differences according to a fixed metric or similarity measure. Patterns in the same class should be similar to one another and different from patterns in the other classes. To distinguish this problem from the pattern recognition problem discussed above, it is common to refer to solution methods as performing "unsupervised learning" or "learning without a teacher". Unlike pattern recognition problems, clustering does not require training sets of patterns; that is, it does not require this explicit "teacher" in the environment. A more accurate way of viewing the clustering problem, however, is to consider it a

pattern recognition problem with the teacher included as part of the adaptive system. Classification errors are computed by the system itself based on the metric or similarity measure. In this sense, clustering is learning with a particular built-in teacher rather than learning with an arbitrary external one.

According to the distinctions we are using, the clustering problem is characterized in the same manner as the pattern recognition problem: 1) open-loop, 2) multiple control situations, 3) partial control surface knowledge, and 4) error-correction. Unlike the pattern recognition problem, however, clustering is more clearly open-loop. Any closed-loop formulation will involve a loop through a part of the environment whose properties are completely known from the start. In fact, the use of the terms control situations and control surface to describe clustering is not very appropriate. Duda and Hart (1973) provide a good discussion of clustering methods.

2.4.3 Stochastic Approximation Methods

An important type of algorithm used to solve pattern recognition problems is known as a stochastic approximation algorithm. Stochastic approximation algorithms are designed to extremize functions that can only be evaluated in the

presence of noise. Variants exist appropriate for both extremum search and error-correction problems, and it is the latter which applies to pattern recognition problems. The adaptive system's response to input (X_a, Z_a) is a function of X_a and a vector of internal parameters $W = (w_1, \dots, w_n)$. To each vector W can be assigned a measure of the classification error that would occur if W determined the classification of all of the training patterns X_a , $a=1, \dots, k$. This measure is usually taken to be the expectation over all input patterns of the sum of the squares of the differences between actual and correct classification (mean square error). Each trial (X_a, Z_a) provides only the error in classifying the single pattern X_a and thus does not provide an exact value of the function to be minimized. Stochastic approximation methods are applied by viewing each individual error as a noisy measurement of the actual error function. They use equilibrium-seeking negative feedback procedures based on these sample errors for individual trials. The theory of stochastic approximation examines conditions under which gradient descent based on sample errors can lead to minimization of the expected error over all patterns. See Duda and Hart (1973) and Kasyap, Blaydon, and Fu (1970), for more complete discussion of this theory. In Section 4 we point out that the Widrow-Hoff learning rule, which is essentially identical to the Rescorla-Wagner model of classical

conditioning, and the perceptron learning rule are both examples of stochastic approximation methods.

2.4.4 The Perceptron Learning Rule

It is instructive to examine some of the details of stochastic approximation methods as they apply to pattern recognition problems. As a representative example, we focus on the fixed increment perceptron learning rule. All of our observations also apply to the Widrow-Hoff learning rule as well as other examples of stochastic approximation methods. Figure 2.3 shows an environment E and an interacting mechanism consisting of three components that together implement the perceptron learning rule: a comparator, an adjustable classifier, and a weight adjustment rule. Different views of this interaction can be obtained by successively considering the dashed lines A, B, and C as boundary lines between an environment and an adaptive system.

Boundary A. All of the components below line A comprise a perceptron as it is sometimes viewed. It is clear that this view shows the perceptron in an open-loop interaction with E. The problem it attempts to solve is the open-loop view of the pattern recognition problem described above.

Boundary B. The most common view of the perceptron results

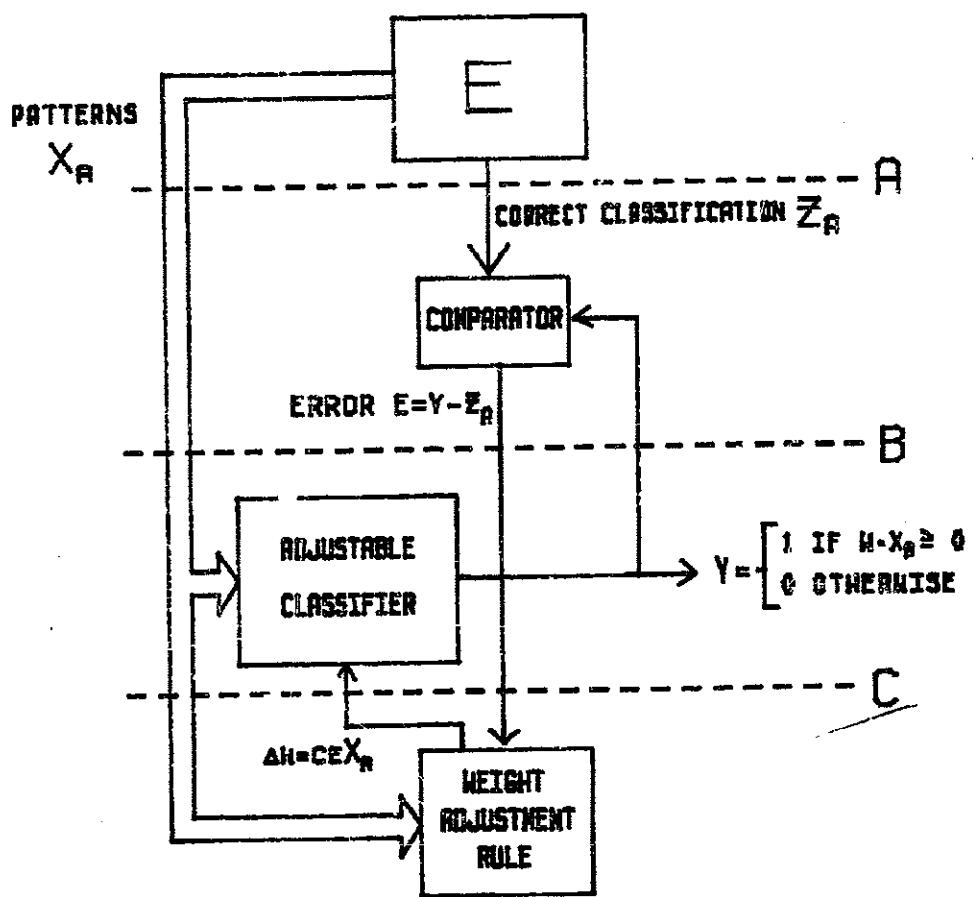


FIGURE 2.3. An analysis of the perceptron learning rule. The dashed lines represent several different ways of viewing the perceptron. See text for explanation.

from considering it to consist of those components below line B interacting with an environment consisting of E and the comparator. According to this view, the perceptron operates as follows: a pattern is presented, and the perceptron computes its classification decision. The environment then provides a signal that is 0 if the classification was correct, +1 if it was 0 but should have been 1, and -1 if it was 1 but should have been 0. According to this view, the perceptron is attempting to solve a closed-loop problem, and it is our belief that many believe this problem to be the prototypical closed-loop learning problem.

It should be clear from our discussion, however, that it is only a very restricted type of closed-loop problem. Indeed, it is characterized by complete a priori knowledge about the nature of the closed-loop interaction, and can be solved by conversion into an equivalent open-loop problem simply by considering the comparator as part of the adaptive system (boundary A in Figure 2.3). Moreover, no additional information is required in order to perform this conversion. If it is known that the closed-loop form of the perceptron learning rule can perform meaningfully when interacting with an environment, then it is also known that the environment contains a comparator that determines the appropriate error signals. This comparator can simply be regarded as a part

of the adaptive system. It makes little difference whether the subtraction is done by the environment or by the perceptron. Therefore, despite common belief to the contrary, we consider the problem that the perceptron learning rule is capable of solving to be an open-loop problem.

Boundary C. Finally, the weight adjustment rule can by itself be regarded as an adaptive system interacting with an environment. Its task is to minimize the error signal by sending appropriate control signals to its environment which consists of the original environment E, the comparator, and the adjustable classifier. The resulting problem can be classified as follows: 1) closed-loop modulatory control, 2) multiple control situations (determined by patterns X and error signal values e), 3) total knowledge of the control surface (its action is always $c\epsilon X$), and 4) error-correction. This problem is closely related to the problem a servomechanism can solve. If X were fixed, for example, the weight adjustment rule would act exactly like a thermostat or a governor. Its task is to control the weights so that a correct classification is made. The classification signal Z acts as the set-point of the servomechanism. Like a servomechanism, its control surface is completely specified based on a priori knowledge and implements negative feedback. We regard this component to be the significant part of the perceptron learning rule.

2.4.5 Function Optimization

The most familiar form of the function optimization problem appears in the framework of Figure 2.1 if the input stimulus S to the adaptive system is a scalar valued signal that is directly determined by the action of AS. Without loss of generality, we can say that $S(t) = f(A(t-1))$ where f is a scalar valued function of the space of possible actions. In other words, E appears to AS as a memoryless system that simply implements a fixed function from AS actions to AS inputs. The goal of the adaptive system is to find that action for which f , the payoff function, index of performance, or reinforcement function, has its maximum value or, what is the same thing, to find that action for which a disutility function, cost function, etc. has a minimum value. This problem is also called a decision problem under certainty. The closely related problem with uncertainty will be discussed below.

According to the basic distinctions we have made, the function optimization problem is characterized as follows:

- 1) It is a closed-loop problem, and the closed-loop interaction is essential. Unlike the case of the closed-loop view of the perceptron, the loop cannot be opened unless the adaptive system itself can determine the value of the payoff function for each action. This requires

information that is not available to the adaptive system in a function optimization task. In addition, a function optimization system has direct control over its input. A given system action always produces the same payoff value (assuming the noiseless case).

- 2) It has a single control situation. A solution method may involve components that face several control situations, but since there is only one function to optimize, the overall problem has only one control situation: there is a single best action. Note that since there is direct control over the environment, the availability of other information to the adaptive system, such as some indication of the environment's state, does not make the task easier. Since the environment is memoryless, its state never changes.
- 3) There is partial knowledge of the control surface. For this problem, the control surface is just the mapping that assigns the single control situation to the optimum system action. The object of the optimization problem is to find this action.
- 4) This is an extremum search problem. The problem is not to make a particular input to occur but to make a maximal (or minimal as the case may be) value of the payoff function to occur. In applications the emphasis is on finding the optimal output rather than on causing the optimal input, but the problem is essentially the same. Perhaps the most

important point is that the payoff value is not in general a measure of the error between actual and optimal outputs as it is in the case of the pattern recognition problem. Neither the adaptive system nor its environment need know from the start what the optimal action is.

Since the function optimization problem involves only a single control situation, it is clear that it is a very restricted type of adaptation or learning problem. Notice, however, that any problem with multiple control situations and partial knowledge of the control surface can be cast as the function optimization problem of finding the optimal control surface. In fact, any adaptation and learning problem can be formulated as a function optimization problem. The reason for this generality is that function optimization provides a way to solve any adaptation problem that involves a preference ordering that can be represented by the ordering of a numerical measure. The function optimization abstraction allows one to consider adaptation with respect to any total preference order rather than a particular preference order. How can a problem that we have called very restrictive also be general enough to encompass all adaptation and learning problems?

The answer is that although any problem can be turned into a function optimization problem, to do so may require

ignoring vast amounts of useful information. Transforming a multiple control situation problem into a function optimization problem, for example, requires the information from the environment telling the adaptive system what control situation is currently present to be totally ignored. The only information the adaptive system receives from the environment in a function optimization task at any time is a scalar value of the payoff function. Thus the equivalent optimization problem is a much more difficult problem than the multiple control situation problem. The lack of control situation information makes the search space enormously larger.

Holland (1975) writes:

Much can be learned from adaptive plans in general by studying plans which act only in terms of payoff . . . In particular, plans which receive information in addition to payoff should do at least as well as plans which receive only payoff information. Thus, the efficiency of payoff-only plans . . . sets a nontrivial lower bound on the efficiency of other plans. (p. 26)

We must agree with Holland, but we think that there is a very large gap between this lower bound and the efficiency of adaptive systems that are able to use neutral environmental information. Most function optimization methods rely on some form of gradient ascent which is often described as the method a blind person would use to find the top of a hill. Our question is: Why limit attention to

algorithms that are blind? The elimination of control situation information corresponds to eliminating all sensory input to an animal except that directly signaling reinforcement. It is obvious that the absence of neutral sensory information makes the survival problem much more difficult. We think that the success shown by animals in solving adaptation and learning problems is due not to particularly effective function optimization methods but rather to the availability of sensory information permitting global optimization problems to be decomposed into many simpler optimization problems, each associated with particular constellations of sensory inputs.

2.4.6 Learning Automata

Learning automaton search methods originated in the work of Tsetlin (1971). There are several formulations of learning automaton search methods, but the simplest can be described as follows. At any time t there is a probability distribution over the set of possible actions of the adaptive system, and an action is chosen by sampling according to this distribution. If a reward occurs as a result of the action, or more generally, if the value of a payoff increases, then the probability of that action is increased while the probabilities of the other actions are uniformly decreased by an amount chosen so as to result in a

new probability distribution. The next action is chosen according to this new distribution. If punishment occurs, or if payoff decreases, then the probability of the action just taken is decreased, and the probabilities of the other actions appropriately adjusted upwards. Many schemes have been proposed to update the probability distributions, the simplest being the linear scheme used by Bush and Mosteller (1955). Holland's genetic algorithm's (Holland, 1975) can be viewed as learning automata with particularly sophisticated methods for updating probabilities. A review of the theory of learning automata is provided by Narendra and Thathachar (1974). It has been found that learning automaton methods can be efficient search methods for solving function optimization problems in which little is known about the nature of the payoff function.

Learning automata are also capable of solving problems that resemble function optimization problems but are really quite different. In our classification scheme, these problems appear exactly like function optimization problems, but the adaptive system action does not deterministically determine the environment's response. The action only determines a probability distribution over the set of possible payoffs from which the environment's response is determined by sampling. These problems are sometimes called decision problems under uncertainty. It is clear that the

function optimization task described above is a special case of this problem.

Tsetlin (1971) studied the ability of learning automata to solve this type of problem by studying their interaction with environments he called stationary random media. The payoff could take only two values: 0, interpreted as reward, and 1, interpreted as penalty. A stationary random medium is characterized by an unvarying probability of reward for each possible action of the learning automaton. An automaton acts "expeditently" in a random medium if it chooses actions so that the expectation of reward becomes greater than it would be if the actions were always chosen with equal probabilities. An automaton is "optimally expedient" if this expectation is equal to the largest probability of reward possible with the random medium, that is, it eventually consistently chooses the action that has the highest probability of producing a payoff of 0 (reward). Many simple learning automata were shown to operate expeditently when interacting with random media, and some were shown to be optimally expedient, although later work showed that they were "almost" optimally expedient (see Narendra and Thathachar, 1974).

The random nature of the environment makes the problem nontrivial even when there are only two possible payoffs.

This task is not simply to extremize a payoff function which the environment evaluates for each automaton action (the environment does not implement such a function), but rather to extremize a function defined for sequences of automaton actions; that is, to maximize a measure of cumulative reward or minimize a measure of cumulative penalty. This function, whose domain is technically infinite, cannot be directly controlled by the adaptive system but can only be modulated. Recognizing how this form of modulation differs from that appearing in a typical control problem, decision problems under uncertainty are classified as: 1) closed-loop modulatory control, 2) single control situation, 3) partial control surface knowledge, and 4) extremum search.

This type of problem involves several aspects of adaptation and learning that are of great importance. One is the inevitable tradeoff between the exploitation of current knowledge and the search for new knowledge. The simplest example is the so-called "two-armed bandit" problem which concerns the allocation of trials for a learning automaton having just two actions in a stationary random medium. Holland (1975) discusses the importance of these issues for theories of adaptive systems. Related issues arise when learning automata interact with nonstationary random media. These environments consist of a collection of

stationary random media that switch from one to another according to unknown transition probabilities. Tsetlin (1971) also studied the behavior of various learning automaton algorithms in nonstationary random media, but these problems tend to be mathematically intractable. One interesting result, however, is that there is a relationship between the degree of stationarity of a random medium and the optimal learning rate. The optimal learning rate becomes larger as the environment becomes less stationary (Tsetlin, 1971). It is noteworthy that even though one would think that a nonstationary random medium would in practice provide clues as to what state it was in, to the best of our knowledge learning automata in nonstationary random media have been studied only as single control situation problems.

Another very interesting capability of learning automata is that they are applicable to problems in which the function to be optimized is vector rather than scalar valued, problems Ho (1970) calls generalized control problems. For example, a set of learning automata is capable of finding certain kinds of optimal solutions of games about which no a priori knowledge is available. Each player is a learning automaton interacting with a nonstationary random medium consisting of the other players and the structure of the game. Narendra and Thathachar

(1974) provide a review of these results. We believe these game theoretic results have significant implications for neural modeling that have never been explored, but it would take us too far afield to discuss them here (see Section 8).

2.4.7 Closed-Loop Control

Thermostats and governors are the simplest systems that solve closed-loop control problems. These systems receive from their environment either an error signal indicating the deviation of the environment's state from a desired state, or they receive more general environmental information and compute their own error signal. They function so as to minimize this error. Figure 2.4 shows the basic organization of a closed-loop control system. Here we are only concerned with non-adaptive control problems.

According to the distinctions we have made, closed-loop control problems are characterized as follows:

- 1) They are closed-loop problems, and control over AS input is usually modulatory. As for the case of the perceptron, the comparator may be viewed as a part of the control system or as a part of the environment. Here, however, if the comparator is taken as part of the control system, then the problem remains closed-loop since the loop passes through E

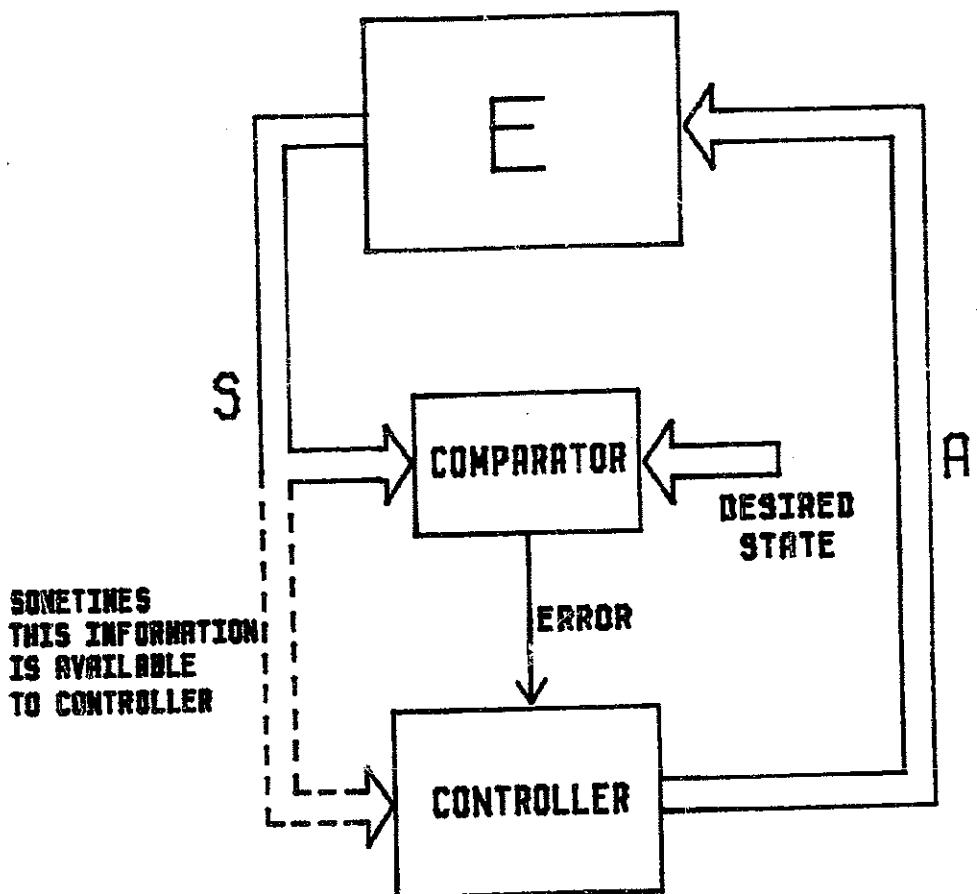


FIGURE 2.4. The basic organization of a closed-loop control system.

and not just the comparator (cf. Figure 2.3).

2) They are multiple control situation problems. The control situation is usually indicated by the value of an error signal but can also depend on other environmental information (as in the case of the weight adjustment rule of the perceptron). The dashed arrow in Figure 2.4 indicates the possible availability of information other than the error signal.

3) There is complete knowledge of the control surface. The control surface is defined by a control law (often characterized by its "gain"), which is fixed from the start, for determining the control system's action based on its input signal. The control law is specified by the control system's designer on the basis of assumed a priori knowledge of environmental dynamics. Consequently, a control system need not search for the correct action. Maintaining the temperature near the desired temperature, for example, is a reflexive function of the thermostat made possible by the suitability of its control law for its environment.

4) They are error-correction problems. The control law is chosen so that the control system/environment composition has the desired equilibrium state. This is accomplished through negative feedback. Although it is possible to view this problem as an extremum search problem, the control rule only succeeds in extremizing a very restricted class of

functions.

2.4.8 Kineses and Taxes

In their classic work The Orientation of Animals: Kineses, Taxes and Compass Reactions, Fraenkel and Gunn (1961) discuss a number of methods used by animals for finding and remaining near light or dark areas, warm or cool areas, or, in general, for approaching attractants and avoiding repellants. They distinguish between two major types of reactions that they call kineses and taxes. A kinesis is a locomotory reaction in which speed of movement and frequency of turning depend on stimulus intensity. A taxis is a reaction in which movement is straight towards or away from the source of stimulation. These terms were proposed to describe animal behavior and are favored over the term "tropism" which originally described a specific class of mechanisms involving conditions of tension in symmetrical muscles (see Fraenkel and Gunn, 1961, pp. 5-10). Recently, Selfridge (1978) provided a view of some of these reactions that emphasizes their general utility as adaptive strategies and has shown how they can be used for improving the performance of artificial adaptive systems.

Of the many different types of kineses and taxes described by Fraenkel and Gunn, we will focus only on those

called klino-kinesis and tropo-taxis. These reactions solve problems that have a particularly interesting place within the range of adaptation and learning problems we are considering in this report.

2.4.8.1 Klino-Kinesis - The most intensely studied example of klino-kinesis occurs in the behavior of various types of bacteria such as Escherichia coli, Salmonella typhimurium, or Bacillus subtilis. These bacteria propel themselves along relatively straight paths by rotating (!) a flagellum. With what at first appears to be random frequency, they reverse flagellar rotation which causes a momentary disorganization of flagellar filaments. This causes the organism to stop almost instantaneously and tumble in place. As the disorganized flagellum continues to rotate in the new direction, its filaments reorganize causing the organism to be again propelled along a straight path. Consequently, flagellar reversal causes a random change in direction of travel.

Adaptively useful behavior results because the frequency of flagellar reversal is modulated by the direction of movement with respect to level of attractants and repellants. Reversal frequency decreases if movement is toward higher attractant concentrations and increases if movement is toward lower concentrations. Repellants have a

similar effect, mutatis mutandis. This modulation of flagellar reversal biases locomotion so that the organism remains near places of maximal attractant concentration or minimal repellent concentration. It is a very effective strategy particularly when gradient information is very noisy. Koshland (1979) describes this type of behavior and underlying biochemical mechanisms in great detail. Selfridge (1978) emphasizes the generality of this type of adaptive strategy, which he calls the Run and Twiddle strategy, by describing it as follows: if things are getting better, keep doing what you are doing; if things are getting worse, do something else. We of course recognize the rationality of this maxim. Yet bacteria use it!

The problem a klino-kinetic or run-and-twiddle strategy solves can be characterized as follows:

- 1) It is closed-loop. The action of the bacterium clearly affects its sensory input in a relevant manner. Importantly, this control over stimuli is modulatory rather than direct. The same action produces different results depending on the location and orientation of the bacterium.
- 2) There are multiple control situations. The control situations are "heading up attractant gradient or down repellent gradient" and "heading down attractant gradient or

up repellent gradient". These are determined by short-term memory which permits the comparison of stimulus levels at successive times.

3) There is complete knowledge of the control surface. The control surface specifies that in the up-attractant-gradient control situation the optimal action is to decrease tumble frequency while in the down-attractant-gradient situation the optimal action is to increase tumble frequency. This knowledge is built-in from the start and constitutes the reflexive kline-kinetic response.

4) This is an extremum search problem. The optimal place in space is not known from the start, negative feedback is not involved, and the organism is incessantly active. The kline-kinetic strategy is not particularly effective for extremizing complex multimodal functions, but this does not mean that it solves an error-correction problem. We suspect that a careful analysis of kline-kinesis would show that it is quite successful for optimizing performance measures that consider long-term cumulative performance in environments with dynamically changing attractant and repellent concentrations; that is, special kinds of decision problems under uncertainty.

We find the problem solved by a kline-kinetic strategy very interesting because it is very similar to a function

optimization problem yet differs from that problem in several crucial ways. The most basic difference is that the kline-kinetic strategy solves a problem involving modulatory rather than direct environmental control. This implies that the payoff function, here the attractant concentration, is not a function of the adaptive system actions. There is no optimum adaptive system action. One cannot say that either raising or lowering tumble probability is best. Which is best depends on the orientation of the organism with respect to attractant and repellent gradients. This is why the problem has multiple control situations while the function optimization problem does not.

But is it not true that the simplest hillclimbing method for function optimization has two control situations, one for up gradient and one for down gradient? This is indeed true, but hillclimbing is a method that can be used to solve a function optimization problem. Kline-kinesis is also a hill-climbing method and can, in fact, be used to solve function optimization problems, but the problem faced by a bacterium is not function optimization as we have defined it. Figure 2.5 helps make these points clear. If the adaptive system AS is taken to consist of the components below boundary A, then it faces a function optimization problem. Each action of this adaptive system is a position in space, and the environment E determines the attractant

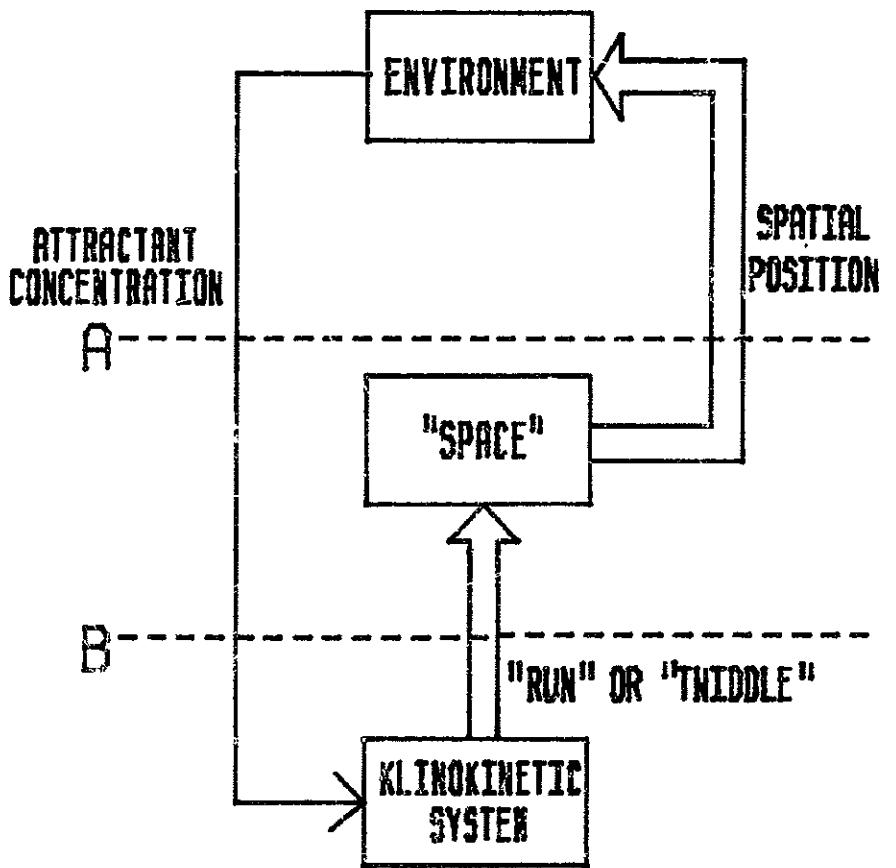


FIGURE 2.5. An analysis of klino-kinesis (or run and twiddle). The adaptive system below line A faces a classical function optimization task. The system below line B, on the other hand, faces an extremizing control problem since it does not have direct control over its input.

concentration for each point in space. A place in space always has the same attractant concentration. On the other hand, if AS is just the kline-kinetic mechanism shown below boundary B, then it faces a different problem. Its actions are 'run' and 'twiddle' (actually they are high and low twiddle probability) which only have a modulatory effect on its input. The component labelled "space" determines new positions based on AS action and current position and orientation. It is accurate to call the problem the kline-kinetic system faces an extremizing control problem since it differs from the standard control problem only by not being an error-correction problem.

2.4.8.2 Tropo-Taxis - A tropo-tactic reaction solves the same problem as the kline-kinetic reaction but in a different manner. Tropo-taxis requires at least two sensory receptors so that the attractant gradient can be detected by means of simultaneous comparison of stimulus intensity received at two different places rather than comparison of intensities received at two different times. In contrast to the almost random appearing paths produced by kline-kinesis, tropo-taxis produces nearly straight, direct paths toward or away from the source of stimulation. Usually this type of reaction is associated with light stimulation, in which case it is often called photo-taxis and, formerly, heliotropism (Fraenkel and Gunn, 1961, p. 76). Our interest in

tropo-taxis lies in the fact that unlike klino-kinesis it incorporates as a component a mechanism that solves an error-correction problem via negative feedback.

For specificity we will discuss positive photo-tropo-taxis (light acts as an attractant). Suppose a continuously moving organism has two light sensitive spots symmetrically placed about the anterior end of its locomotory axis. A prespecified control surface causes turning movements in the direction of the most strongly illuminated receptor. If the receptors are equally illuminated, no turning occurs. The control surface implements a negative feedback error-correction control system that seeks to equalize the stimulation of the receptors. Logically (but not literally in an animal) a signed error is computed by subtracting one receptor activity from the other. The control mechanism causes the organism to have two equilibrium orientations with respect to a light intensity gradient, one facing up-gradient and the other facing down-gradient. Since we are describing positive photo-tropo-taxis, the up-gradient equilibrium is stable but the down-gradient one is not. A very small amplitude random turning component prevents the down-gradient orientation from being maintained. Animals with compound eyes possess more complex control surfaces. There are more than two control situations since stimulation

of posterior receptors causes stronger turning than stimulation of anterior receptors. This yields a more refined orientation control system. Tropo-taxis thus consists of a negative feedback orientation control system coupled with continuous movement. The result is a strategy capable of very efficiently solving simple extremum search problems.

2.4.9 Adaptive Control

A control problem is an adaptive control problem if the control surface is not completely known from the start and can be modified based on the individual experience of the control system. This additional flexibility is necessary for applications in which not enough knowledge of the environment exists to permit the specification of a fixed control law. Adaptive control problems therefore have the same characteristics as the control problems discussed above with the exception that there is only partial control surface knowledge. The problem of determining the control surface through experience is another type of adaptation or learning problem that can take a variety of forms. Adaptive control problems are therefore compositions of the basic problems we are considering.

Figures 2.6 and 2.7 show two general forms of adaptive

control systems. In each case, the component(s) below the dashed line comprise an adaptive system whose actions determine modifications in the control law of the control system shown above the dashed line. In each case, there is a preference ordering determined by a measure, usually called an index of performance or performance criterion by adaptive control theorists. Sometimes the determination of the index of performance involves a system identification procedure. The problem the adaptive mechanism must attempt to solve can have any combination of the basic properties we have been discussing. We call this problem the second-level problem of the adaptive control system.

Figure 2.6 shows an adaptive control system whose second-level problem is an error-correction problem since there is a known desired value of the index of performance. This second-level problem is in fact identical to the non-adaptive control problem discussed above. The second-level control surface is completely known. An example of this type of adaptive control problem is that of determining the appropriate sign for the gain of the adjustable controller. The index of performance may be the same as the error signal to the first-level controller. Imagine the problem faced by a thermostat that can be connected to a furnace in either of two ways: its signal either turns the furnace on or it turns it off. The

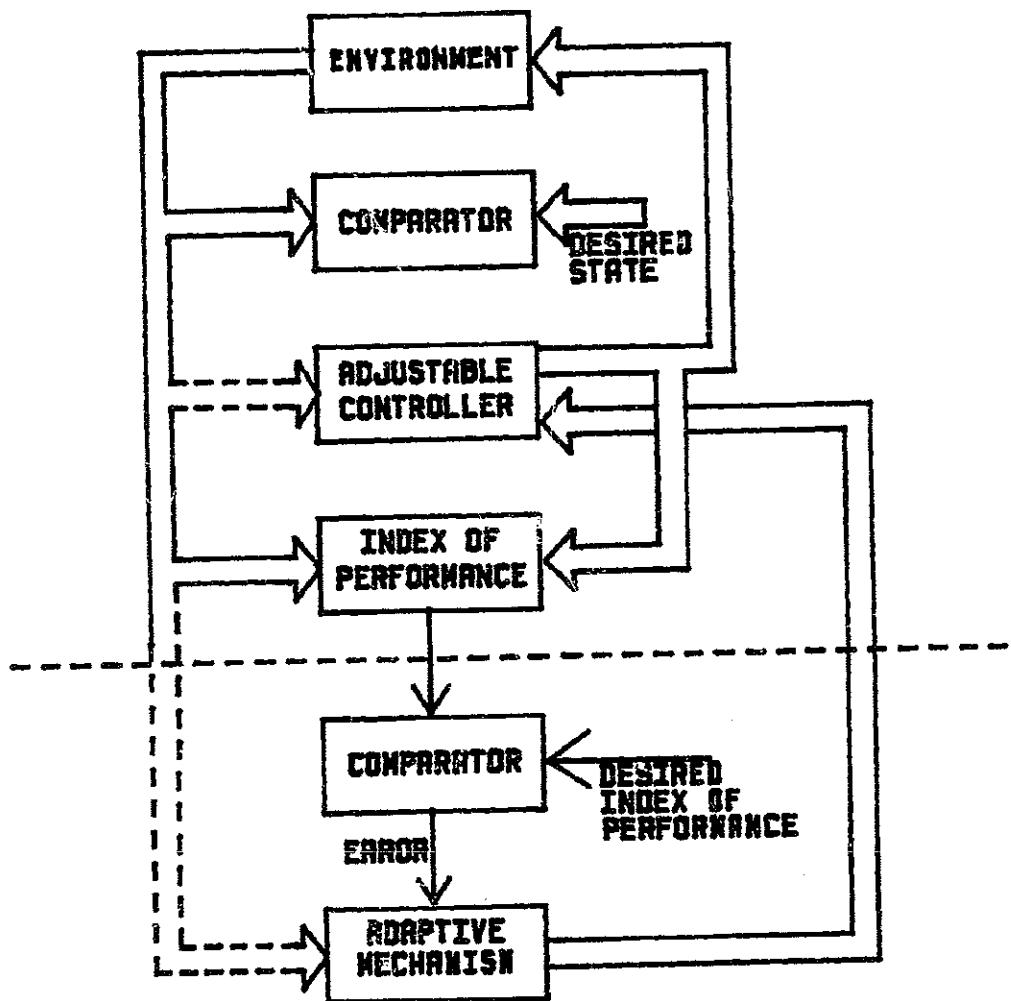


FIGURE 2.6. The organization of an adaptive control system. The part of the system below the dashed line faces the second-level problem of the control problem. Here, this second-level problem is an error-correction problem.

second-level adaptive mechanism must try gains of both signs until the appropriate negative feedback is obtained through the furnace loop. This second-level mechanism must perform an error-correction search and can stop when the correct gain is found.

Ashby's Homeostat (Ashby, 1960) is exactly this type of adaptive control system. The Homeostat's task is to maintain the values of a set of critical variables within prescribed limits. This is a closed-loop control problem. Ashby suggests the use of a second-level "step function" mechanism to alter the values of control law parameters. The rule he suggests for selecting parameters is as follows: if the value of the index of performance is acceptable (that is, all critical variables are within prescribed limits), do nothing; if it is not acceptable, choose any new parameter value. This is the simplest method for solving an error-correction problem in which the error measure only takes the values "acceptable" and "unacceptable". At all levels, the Homeostat is capable of solving only error-correction problems. When the desired state, which is known from the start, is achieved, all activity of the Homeostat ceases.

Figure 2.7 shows another type of adaptive control system known as an extremizing adaptive control system.

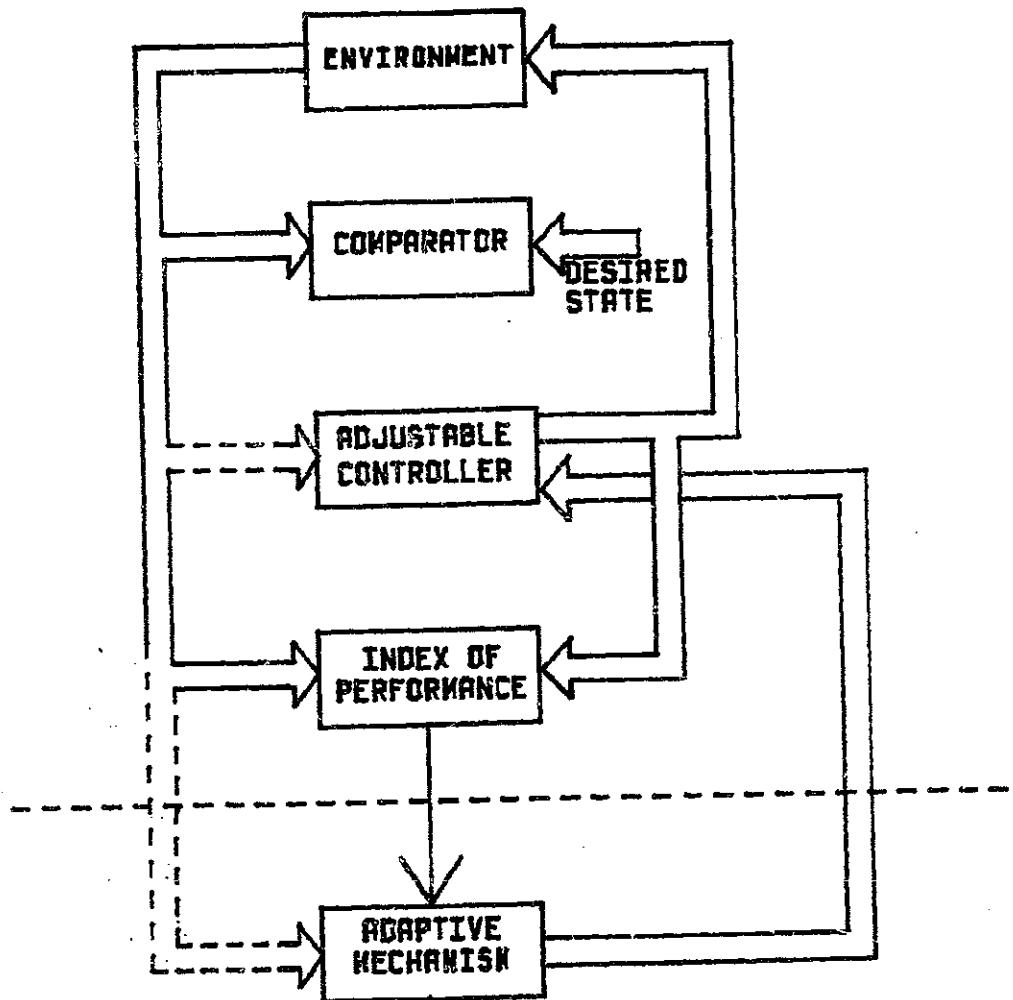


FIGURE 2.7. An extremizing adaptive control system. The second-level problem faced by the component below the dashed line is an extremum search problem.

Here the second-level problem is to minimize or maximize the index of performance (depending on what kind of index it is) with no explicit knowledge of what constitutes the optimal action. In other words, the second-level problem is an extremum search problem. An example of this kind of problem is to control an internal combustion engine in order to maximize combustion efficiency.

Our discussion of adaptive control leaves much unsaid. It is a very large subject, and our few comments merely provide a rough outline of the basic types of problems usually considered. We think the most salient features of adaptive control are:

- 1) The second-level adaptation problem is characterized either by multiple control situations but a completely known control surface (Figure 2.6), or by a single control situation but an unknown control surface (Figure 2.7).
- 2) There is considerable a priori knowledge about the environment in which the second-level adaptive system must operate since it includes the first-level control system and a carefully defined procedure for computing the index of performance.

2.4.10 Learning Control

The distinction between adaptive and learning control seems not to be sharply defined. Mendel and McLaren (1970) describe learning control systems as adaptive control systems with long-term memory. The second-level control law is broadened by "localizing the adaptation to regions in a plant-environment space and by providing the control law with a long-term memory". In our terms, learning control results when the second-level adaptation problem has multiple control situations and partial control surface knowledge. This is easiest to understand through an example.

Ashby (1960) describes an elaboration of the 'Homeostat enabling it to function efficiently in a variety of different environmental situations by the "accumulation of adaptations". Recall that the Homeostat is an adaptive control system whose first-level control law is modified by a second-level parameter selection mechanism until the performance criterion is reached. Consider letting the Homeostat reach equilibrium in one environmental situation and then placing it in a new environmental situation so that new parameters have to be selected by the second-level mechanism. Ashby suggests that if the Homeostat is given long-term memory so that it can recognize a situation in

which it had previously adapted, then the appropriate parameter values can be immediately retrieved without the error-correction search. A search is required only in novel environmental situations.

According to our terminology, the second-level adaptation mechanism in the adaptation accumulating Homeostat is characterized by multiple control situations but, unlike the ordinary Homeostat, by partial knowledge of the control surface. Further, the control situations are determined not just by an error signal but also by signals indicating salient features of the environment. The control surface is the long-term memory that is filled in as adaptations "accumulate". The entire characterization of this problem is as follows: 1) closed-loop, 2) multiple control situations, 3) partial knowledge of control surface, 4) error-correction.

Ashby does not describe in detail how the adaptation accumulating Homeostat can recognize control situations, but it is clear that some form of pattern recognition is required. Pattern recognition is, in fact, an intimate part of any problem with incomplete control surface knowledge.

The extremizing counterpart of the type of learning control problems solved by the adaptation accumulation

Homeostat is characterized as follows: 1) closed-loop, 2) multiple control situations, 3) partial control surface knowledge, 4) extremum search. It therefore requires a multiple control situation extremizing search procedure. Mendel and McLaren (1970) discuss problems of this type which they call reinforcement learning control problems. Figure 2.8 shows a representative reinforcement learning control system. They say that the following procedures are required to automatically improve the (first-level) control law: 1) the goal circuit evaluates the results of previous control choices of the learning network for given situations, and 2) the learning control law's memory is modified so that subsequent control choices reflect this evaluation (after Mendel and McLaren, 1970, p.295).

An example of a reinforcement learning control system is the BOXES system of Michie and Chambers (1968). We describe it in some detail since it illustrates the features required in a learning control system in a particularly simple and elegant form. The control task chosen for demonstration purposes is to balance a pole on a cart that can move along a track of fixed length. The control system can send signals causing the cart's motor to exert full force either 'left' or 'right' for a fixed duration. At regular time instants the control system receives signals from the cart-pole apparatus consisting of four element

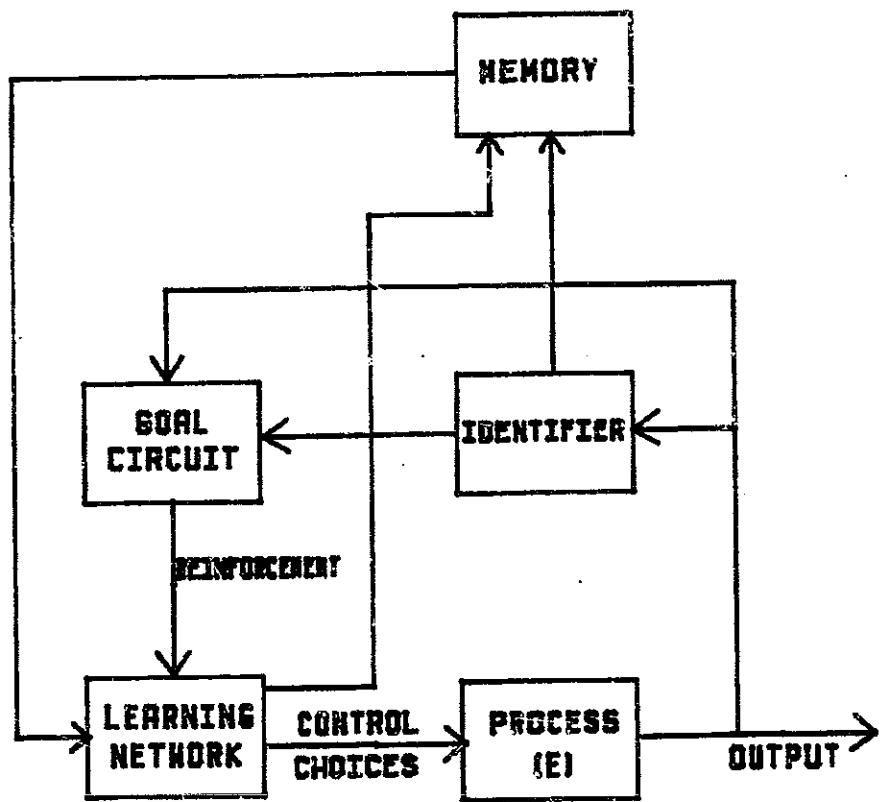


FIGURE 2.8. A representative reinforcement learning control system (after Mendel and McLaren, 1970, Figure 2, p. 295). See text for explanation.

state vectors (x , \dot{x} , θ , $\dot{\theta}$) whose elements respectively give the cart's position and velocity and the pole's angle and the rate of change of the angle. In addition to these signals, a 'failure' signal is sent to the control system whenever the pole falls or the cart runs off the track. The system's task is to construct a mapping from the state vectors to the system actions in such a way that the occurrence of the failure signal is minimized. This mapping is the control surface.

A long-term memory is provided to accumulate the control surface knowledge. The space of all possible state vectors is quantized by distinguishing only three grades of position x , three of velocity \dot{x} , six of angle θ , and three of angle-change $\dot{\theta}$. This results in a long-term memory having 162 "boxes" each corresponding to a rather coarse region of the state space. The problem is to appropriately store 'left' or 'right' in each of these boxes.

The method used by BOXES is very simple. Associated with each memory box is a mechanism that chooses an action whenever the box is "addressed" by an environmental state vector and accumulates a record of time-until-failure for each action. In particular, the mechanism for the box addressed by the current state does the following:

- 1) Chooses 'left' or 'right' depending on which is expected

to lead to the longest time until failure.

- 2) Remembers which action is chosen and initiates a count.
- 3) When failure occurs, the count is used to update either the left or right expectation depending on which action was chosen.

Michie and Chambers remark that this control scheme really works, and that the control task is nontrivial.

The BOXES system illustrates in a very clear manner the general features of how long-term memory is used in a learning control problem. Mendel and McLaren (1970) distinguish short-term memory from long-term memory by saying that short-term memory records information only for as long as the system is in the same control situation, whereas long-term memory records information that can be retrieved outside the control situation or when the same control situation is entered at a later time. Long-term memory is essential for accumulating control surface knowledge. Mendel and McLaren also point out that long-term memory is also essential for recording information required to construct the control surface (e.g., the expectation records of the BOXES system). If learning cannot be completed the first time a control situation is entered, long-term memory is required to store certain kinds of information so that learning can continue when the control surface is entered again. The associative search network

described in Section 5 is an example of a reinforcement learning control system that is very similar to the BOXES system but that uses a distributed rather than a localized memory.

2.4.11 Instrumental Conditioning

The distinction we have made between problems and mechanisms provides a convenient way of handling some controversial questions regarding what happens in classical and instrumental conditioning experiments. By virtue of the experimental design, an animal's interaction with its environment is different in classical and instrumental conditioning experiments. Assuming that a classical conditioning experiment is characterized by a lack of response contingency (a feature very difficult, if not impossible, to enforce in practice), instrumental conditioning experiments are distinguished from classical conditioning experiments by the fact that they involve response contingencies permitting the animal to exert a degree of control over its input. With this fact there is no disagreement.

Recall, however, that the possibility that a system can control its input does not imply that the problem it solves, or attempts to solve, is closed-loop. It has been a matter

of considerable controversy among animal learning theorists whether or not, to use our terminology, the problems an animal seems to solve in an instrumental conditioning experiment are different from those solved in classical conditioning experiments. Is the control over input used in any way by the animal? It is now generally agreed that such control can make a behavioral difference (see, for example, Dickenson and Mackintosh, 1978). Consequently, it is safe to say that unlike the problem faced by an animal in a classical conditioning experiment, the problem faced in an instrumental conditioning is closed-loop, or, at least, is closed-loop for most instrumental paradigms. Nevertheless, it is problematic to speak of classical or instrumental learning rather than the behavior elicited in classical or instrumental experiments.

We think it is also fairly safe to consider the problems typically solved by animals in instrumental conditioning experiments as multiple control situation problems. For some experimental paradigms the control situations are explicitly signaled by, for example, a discriminative stimulus signaling the availability of reinforcement. In other experiments, they may be less explicitly signaled and can depend on such things as the time elapsed since the last reinforcement event. Additionally, the sensory input from the entire experimental

situation, including the sights, sounds, and smells of the experimental apparatus and experimenter, can act as signals potentially specifying control situations. It is also not misleading to include information about internal motivational states as potentially signaling control situations.

The multiple control situation nature of instrumental conditioning problems was expressed clearly by Thorpe (1951):

The essence of trial-and-error learning [type II or instrumental], then, is the development of an association, as the result of reinforcement during appetitive behavior, between a stimulus or situation and an independent motor action as an item in that behavior when both stimulus and motor action precede the reinforcement and the motor action is not the inevitable inherited response to the reinforcement. (Thorpe, 1951, p. 78)

Multiple control situations are implied by the mention of the development of associations between stimulus situations and motor actions.

Thorpe's description also plainly indicates that there is partial knowledge of the control surface. Both the phrases "independent motor action" and "the motor action is not an inevitable inherited response" mean exactly that there is partial control surface knowledge. An example of a dependent, inherited response is the reflexive response of a

servomechanism such as a thermostat to the error signal.

Finally, we turn to the question of whether or not instrumental learning problems are extremum search or error-correction problems. Do animals tend, for example, to maximize reward, or do they tend to control reinforcement rates toward certain known desired values? The most common interpretation of the data is that animals tend to maximize reward and minimize punishment, but this really involves complex issues that we cannot adequately address here. We will tentatively accept this extremizing view as being consistent with the data if it is noted that animal behavior seems to be appropriate for solving extremum search problems under uncertainty.

Keeping our qualifying remarks in mind, we can characterize the behavior elicited in instrumental experiments as follows: 1) closed-loop, 2) multiple control situations, 3) partial control surface knowledge, and 4) extremum search under uncertainty. This is the same type of problem faced by the second-level adaptive mechanism in a reinforcement learning control system. It is significant that although this problem is related to the problem solved by the perceptron learning rule, it is also very different. It is our impression that the perceptron learning rule, especially in its closed-loop form, has been considered by

some to be capable of solving reinforcement learning problems. This is simply not the case. It is also clear that instrumental conditioning problems are not simple function optimization problems since they involve neutral sensory input signaling multiple control situations. We have found that the area of adaptive system theory most closely related to instrumental conditioning is that of reinforcement learning control systems as discussed by Mendel and McLaren (1970).

2.4.12 Klopff's Heterostat

Klopff (1972, 1979, 1981) proposed a learning rule, in the form of a postulate about synaptic plasticity, which is actually best seen as a learning rule capable of solving simple reinforcement learning control problems. He hypothesized that neurons try to maximize their level of membrane depolarization by changing synaptic effectiveness in the following way: Whenever a neuron fires, those synapses that were active during the summation of the potentials leading to the discharge become eligible to undergo changes in their transmission effectiveness. If the discharge is followed by further depolarization, then the eligible excitatory synapses become more excitatory. If the discharge is followed by hyperpolarization, then eligible inhibitory synapses become more inhibitory. In this way a

neuron will become more likely to fire in a situation in which firing is followed by further depolarization and less likely to fire in a situation in which firing leads to hyperpolarization.

The term heterostat was chosen to emphasize the difference between this hypothesis and those suggesting that the concept of homeostasis plays the central role in understanding the purposiveness of living organisms. Rather than acting solely to achieve a condition in which certain variables remain within particular bounds as suggested by Ashby (1960), a heterostat acts so as to extremize the value of a particular variable. Our discussion of kineses and taxes should make it clear that natural adaptation mechanisms do indeed involve more than equilibrium-seeking or error-correction. Kineses and taxes are ubiquitous in nature and solve extremizing rather than error-correcting problems. We wholeheartedly agree with Klopf's claim that the identification of adaptive behavior with equilibrium-seeking behavior is very misleading. We shall see, however, that Klopf's heterostat is more complicated than the extremizing counterpart of Ashby's Homeostat. It is, in fact, the extremizing counterpart of Ashby's adaptation accumulating Homeostat together with a pattern recognition mechanism.

According to the distinctions we have been focusing upon in this report, the problem various formulations of the heterostat can solve can be characterized as follows:

1) It is a closed-loop problem. Since a pathway becomes eligible for modification only when a presynaptic signal causes a response from the postsynaptic element, the process is closed-loop. The consequences of the system's actions are used to alter long-term memory. The closely related learning rule we study in Section 4 as a model of classical conditioning is open-loop since postsynaptic response is not necessary to trigger eligibility. Of course, either model can be placed in an environmental interaction in which control can be exerted over input, but the presence of the output contingency of Klopff's original proposal permits a heterostat to use the control over its input.

2) There are multiple control situations. As in the case of the perceptron, control situation information is provided by the input signals. For its simplest formulation, the heterostat has two possible actions. Consequently, there are two control situations: one in which the optimal response is 1 and another in which the optimal response is 0. Part of the task accomplished by the heterostat is the classification of input patterns according to which control situation they signal. This part of the heterostat's behavior is similar to perceptron pattern recognition

behavior with the crucial difference that which control situation an input pattern signals need not be explicitly indicated by the environment either directly or by means of an error signal.

3) There is partial knowledge of the control surface. As in the case of the perceptron, the control surface is specified when the weights reach the values that cause the optimal action to be performed in response to each input pattern.

4) This is an extremum search problem. A search is required to find the optimal action for each pattern. By optimal is meant that action which is followed by the largest increase in reinforcement (or by the smallest decrease in reinforcement if only decrease is possible). This is in sharp contrast to the meaning of optimal in the case of the perceptron. Additionally, some forms of the heterostat can perform extremum search under uncertainty. In this respect, the heterostat is closely related to the learning automaton search methods described above. Unlike learning automaton methods, however, the output probability distribution depends on the current input pattern. Thus, as the weights change, the mapping from input patterns to output probability distribution changes. It is not misleading to describe this version of the heterostat as a collection of learning automata together with a pattern recognition scheme. The Associative Search Network described in Section

5 uses this type of adaptive element.

According to this classification, it is correct to say that a heterostat is a simple but complete reinforcement learning control system. In addition, the heterostat as originally suggested by Klopf has the capability of modifying its preference ordering of inputs. Preference is determined by a measure of reinforcement which is, in the simplest case, dependent on a weighted sum of the input signals. But since the weights are modifiable, the preference ordering is modifiable also. For example, as the weight of a particular pathway increases, the relative preference for input patterns with signals over that pathway increases. Control over preference order or performance measure is a very important aspect of adaptive behavior. We think that in the case of the heterostat this capability can be used to construct more informative reinforcement signals from initially neutral environmental information. This capability, which seems closely related to the notion of secondary reinforcement in animals, is not understood well enough at this time to permit us to provide a thorough analysis.

2.5 Summary and Discussion

- 1) The perceptron learning rule and similar stochastic approximation methods such as the Widrow-Hoff rule solve

open-loop problems. Although it is common to view these methods as solving closed-loop problems, the nature of the loop is known from the start to merely pass through a comparator. These problems are equivalent to open-loop problems. They are also error-correction problems. Even putting aside the usual objection that perceptrons can only implement linear discriminant functions, it is clear that they can solve only a very restricted kind of problem. They are not adequate models of animal behavior in instrumental conditioning experiments. Perceptrons are more closely related to classical conditioning.

2) The function optimization problem is a genuine closed-loop problem. However, it is misleading to view all adaptation and learning tasks as function optimization tasks. Since a function optimization procedure is assumed to have direct control over its environment, the function optimization problem is characterized by a single control situation. In other words, the environment is assumed to remain in a single state and implement a single memoryless function of the action choice of the optimization procedure. This implies that there is a single optimal action (or a set of actions with equal optimal payoffs), and that environmental information other than payoff function values is irrelevant.

It is indeed true that any of the problems we have

discussed can be viewed as the function optimization problem of finding the optimal control surface. But this view necessitates ignoring the vast amount of useful environmental information signaling control situations. By definition, function optimization methods are blind to information other than payoff information. It is obvious that the availability of other information can make the search for the optimal control surface much easier. While certain important aspects of adaptation and learning are captured by the function optimization formalization, it is clear that the identification of adaptation or learning with function optimization is misleading.

3) It remains a popular view that the goal-seeking behavior of organisms can be equated to the equilibrium-seeking behavior of servomechanisms. Ashby's theory of adaptation has done much to perpetuate this view. For example, in Design for a Brain (1960) Ashby states:

We can now recognize that 'adaptive' behavior is equivalent to the behavior of a stable system, the region of stability being the region of the phase-space in which all the essential variables lie within their normal limits. (p. 64)

And further:

The point of view taken here is that the constancy of the essential variables is fundamentally important, and that the activity of the other variables is important only in so far as it contributes to this end. (p. 67)

This view of adaptation provided by the early cyberneticists

did much to de-mystify the nature of so-called "teleological" behavior observed in animals. However, this view of adaptation is very restricted in scope.

Klopf (1972, 1979, 1981) has suggested that it is more accurate to view adaptation and learning as, to use our terminology, extremum search rather than error-correcting. According to Klopf's theory, the constancy of some variables is important only in so far as it contributes to the extremization of others. This is exactly the reverse of the view put forward by Ashby. Our investigation has led us to agree that, in a logical sense, extremizing behavior is more fundamental than error-correction behavior. The reason is simply that error-correction problems are restricted types of extremum search problems. If one has a device capable of solving even relatively simple extremum search problems, then one also has a device that can solve any error-correction problem (albeit with some loss of efficiency). But an error-correction device can solve only a very restricted class of extremum search problems. Herein lies the fundamental importance of Klopf's theory: it is the first theory of neural plasticity to consider less restrictive types of extremum search problems than those arising from error-correction problems.

Which kind of problem is more fundamental in nature,

rather than logically, is a different question that is more difficult to answer. It is very clear, however, that many of the prime examples of adaptation in nature involve extremum search. The evolutionary process itself is the expression of a complex adaptation mechanism that is clearly not error-correcting. Nowhere is there knowledge about what base pairs of DNA code for the optimal organism! Klino-kinesis and tropo-taxis, commonly found in nature, are both extremum search methods. They attempt to maximize levels of attractants and/or minimize levels of repellants. A tropo-tactic mechanism contains a mechanism that solves an error-correction problem, but it is clear that the equilibrium attained serves only to facilitate the extremization process. More than this needs to be said about which is more fundamental in nature, extremum search or error-correction. One could argue, for example, that a klino-kinetic or tropo-tactic strategy is used by an organism in order to maintain nutrient intake within acceptable limits, an error-correction problem. We think that views suggesting the primacy of homeostatic mechanisms neglect the fact that animals evolve in very competitive environments having limited resources. Extremum-seeking components in an animal's control system may permit survival in a wider range of environmental conditions than would be possible with error-correction mechanisms alone. What is certain, however, is that one cannot restrict attention to

error-correcting processes as suggested by Ashby and other early cyberneticists.

A FINAL REMARK

Much of the criticism of the approach to developing intelligent systems based on numerical, data directed methods (typified by the perceptron learning rule) rested on the difficulty in extending these methods to solve more difficult examples of the types of problems they were already solving. For example, implications of the perceptron's limitation to forming linear discriminant functions were pointed out by Minsky and Papert (1969), and the shortcomings of hill-climbing methods for the optimization of functions with large plateaus or many false optima were pointed out by Minsky and Selfridge (1960). The criticisms we have implicitly made in this report are of a completely different kind. We have pointed out the restricted nature of the problems these methods were designed to solve rather than their limited ability to solve them. We, of course, agree that general pattern recognition and function optimization problems are very difficult to solve completely, but we think problems of this difficulty need never occur. Pattern recognition is usually just one part of a complex adaptation or learning task, and the function optimization task is so abstract that the formulation of a problem as such a task usually requires

potentially valuable structure and information to be ignored. It seems to us that sophisticated adaptive behavior can result from a system designed to solve a variety of interrelated adaptation and learning tasks, each of which is relatively simple. In other words, when formulated in an appropriate manner, sophisticated adaptative behavior need not require any single subsystem to form highly nonlinear discriminant functions or optimize functions having broad plateaus or many extrema.

SECTION 3

EVOLUTION OF HETEROSTAT MODELS

3.1 Introduction

In this section we present a roughly chronological trace of heterostat models that we have considered during the contract period, noting what we view as key advantages and failings of each. To give an overview, the models can be divided into three groups or periods, which correspond roughly to the chronological progression. The first period was one of wide ranging exploration and experimentation, ending with a conversion from using input level to the adaptive element as reinforcement [footnote], as Klopf originally suggested, to using change in input level as reinforcement. In the second period, the model was further refined, and a number of variations considered. In the

The word "reinforcement" in this chapter is used in a generic sense that includes both reward and punishment.

third period, which overlaps with and in no way supersedes the second, we have begun to consider models that use a specialized reinforcement pathway whose activation does not affect the activity of the element. The early stages of the third period stretched over a long period of time as we gradually came to realize that many of the most interesting aspects of Klopf's heterostat did not require the additional novelty of generalized reinforcement, that is, the ability of all or many input pathways to provide reinforcement.

The main purpose of this section is to record each of the major steps in the evolution of our element designs, as well as to present what we view as the major reasons for each step. The purpose is not to justify and defend each step, for that would be at least as ambitious a project as this entire report. In other sections we more carefully justify particular elements with reference to particular learning tasks (Sections 4, 5, and 6).

3.2 Early Models: Open-Loop Stability

The early period was one of wide ranging exploration and experimentation. A great many models were generated, and we gradually came to some understanding of the fundamental problems involved in creating a workable heterostat. In describing this period, we have chosen to

present relatively few models while putting considerable emphasis on the principles involved. The major advance of this period was a gradual recognition of the advantages of switching to a reinforcement measure based on the change in input to the adaptive element rather than on the absolute level of the input.

Klopf did not provide a complete formal specification of his original heterostat (Klopf, 1972, 1979, 1981). The primary differences among early attempts to interpret his work involved the eligibility and zeroetting mechanisms. The "weighted correlation model with zeroetting" of the fall of 1977 shows how the heterostat was formalized near the beginning of the contract period.

3.2.1 Weighted Correlation Element with Zeroetting

See Figure 3.1 for a diagram of the model's parts and a summary of the notation.

- a) Discrete time, $t=0,1,2,\dots$
- b) Output at time t : $y(t)$
- c) Input at time t : $x_i(t)$, $i=1,\dots,n$
- d) Each synapse i , $1 \leq i \leq n$, has weight $w_i(t)$ at time t
 - excitatory: $0 \leq w_i(t) \leq W_{\max}$
 - inhibitory: $-W_{\max} \leq w_i(t) < 0$

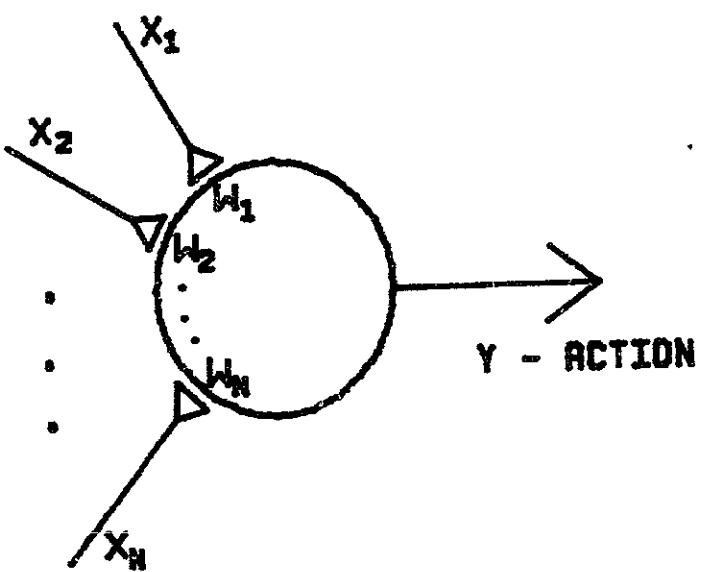


Figure 3.1. Notation diagram for simple heterostat.

e) Weight changes given by:

For excitatory synapses:

$$w_i(t+1) = F\{w_i(t) + E_i(t) \cdot [\sum_{j=1}^n w_j(t)x_j(t)]\} \quad (3.1)$$

For inhibitory synapses:

$$w_i(t+1) = -F\{w_i(t) - E_i(t) \cdot [\sum_{j=1}^n w_j(t)x_j(t)]\} \quad (3.2)$$

where F might be given by:

$$F(x) = \begin{cases} 0 & \text{for } x < 0 \\ W_{\max} & \text{for } x > W_{\max} \\ x & \text{otherwise} \end{cases} \quad (3.3)$$

These rules imply that excitatory and inhibitory weights will remain non-negative and non-positive respectively.

f) Each synapse i has state $E_i(t)$ of eligibility at time t

with $0 \leq E_i(t) \leq 1$ and

$E_i(t)$ computed by a system with memory:

$$E_i(t) = \sum_{k=0}^M f(k)x_i(t-k)y(t-k) + \sum_{z=0}^M h(z)x_i(t-z) \quad (3.4)$$

The functions f and h may look respectively something like those shown in Figure 3.2a and Figure 3.2b.

The zerosetting mechanism is intended to work as follows: Say that $f(k)$ is maximum at $k=T$. A maximum eligibility for synapse i requires that its activity was high and output level was high around T time steps ago and between that time and now its own activity was low. The second sum will produce a large negative number if x was highly active in the interval from now back to T time steps ago. This decreases eligibility.

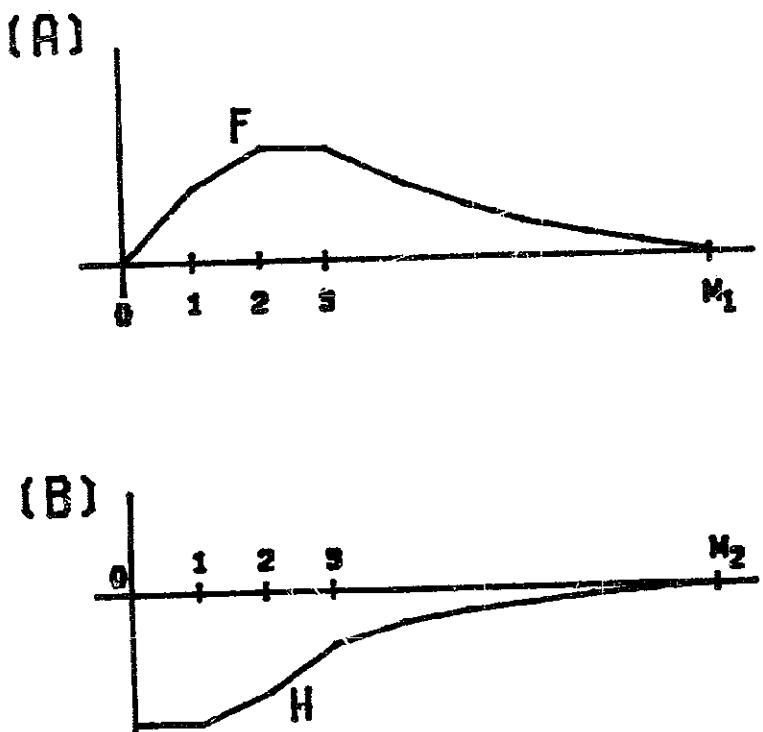


Figure 3.2. (A) An inverted-U shaped eligibility "kernel." (B) A "kernel" function used in computing one form of zerosetting.

Note that in this model all synapses are allowed to decrease as well as to increase in transmission efficacy. Along similar lines, in most later models, synaptic efficacies can change sign, from excitatory to inhibitory or vice versa, a property difficult to defend physiologically.

Representing each synaptic efficacy as a real valued weight that can change arbitrarily is the simplest possibility mathematically, but is problematic insofar as neurophysiological support is desired. We justify our concentration on mathematically simple models on the basis of the insights that mathematical tools may provide into the abstract problems of adaptation and inter-element cooperation. We feel confident that these mathematical models can be mapped back into physiologically plausible neural analogs (although not always mapping each adaptive element into a single neuron), and we have suggested possibilities for some models (e.g., see Section 4 and Barto and Sutton, 1980). We think that the assumptions we have made for the sake of mathematical simplicity (bi-directional, sign changing synapses) have not changed the character of these adaptive systems in any fundamental way, but further work will be required to resolve these issues definitively.

3.2.2 Open-Loop Stability

The major concern during the early period of heterostatic element exploration (1977-78) was the open-loop stability of the adaptive element. A fundamental part of the heterostat concept is that such a device should detect and utilize the effect of its output on its subsequent input; that is, the emphasis is on the closed-loop nature of a device's relationship to its environment. In closed-loop adaptation tasks, great care must be taken to keep the adaptive system from becoming unstable and snowballing into an adaptively useless state. Most of the early heterostats, however, were unstable even in the open-loop case; that is, the case in which their input was completely independent of their actions. Although a closed-loop adaptive element is helpless in a real sense in an open-loop interaction, instability in this case still seemed to be an inappropriate response. Open-loop stability became an important heuristic filter that we applied to adaptive element proposals.

The open-loop instability of early elements was due to their use of excitation and inhibition as reward and punishment respectively. There was no guarantee that an element would, on the average, receive equal amounts of excitation and inhibition. For example, if a neuron

operated according to this principle, then if membrane potential departed most frequently from the resting potential in a certain direction, then all synaptic efficacies of the neuron would move relatively rapidly in that direction until they reached their maximum values (Sutton, 1977). Moreover, the speed at which this process would take place for a single neuron would be independent of the number of neurons or synapses in the brain. Large numbers of neurons would degrade just as quickly as a few.

One solution to this problem would be to measure reinforcement from its average value. In this way, the "effective reinforcement," and thus the weight changes, are prevented from being overwhelmingly either positive or negative. However, there is still no meaningful bound on the weights. Each time a neuronal action potential is followed by greater than average excitation the relevant synapses would be pushed higher until they reached their limits, irrespective of the size of the increase in excitation. The result would be a great reduction in the sensitivity of the adaptive element, since an element employing this scheme could record the sign of reinforcement following firing but could not distinguish relative magnitudes. The solution to this problem was found to be the introduction of an internal negative feedback loop controlling the weights - the larger the weight, the greater

the reinforcement increase necessary just to maintain its level after each use (cf. Uttley, 1979; see Section 4.5.4). This addition caused each weight to asymptotically approach a level proportional to the reinforcement increase it predicted (Sutton, 1978a). However, a local, open-loop instability still existed. An increase in excitation as reward increased synaptic efficacies, and this in turn resulted in further increases in excitation, and so on:

To show how the instability arises, consider a neuron with many synapses. Assume the presynaptic neurons of these synapses fire in a totally random way with a fixed probability distribution. Also assume the initial average algebraic sum of input (reinforcement) to the neuron is zero (although it undergoes random fluctuations of course, depending on which of the presynaptic neurons happen to be firing and transmitting signals through their synapses. Consider what happens if the neuron fires and then, by chance, the reinforcement (input) following the firing happens to be slightly positive. Since this reinforcement is positive it will tend (in most cases) to make the synapses which caused the firing more positive if they were excitatory and less negative if they were inhibitory. In general, the positive reinforcement will result in changes to the synapses which will cause average input subsequently to be slightly higher than it was before, or in this case, slightly positive. Thus, when the neuron fires again it will probably get slightly positive input, which will cause new synaptic increases and thus further raise the level of average reinforcement. This process accelerates until it is completely irreversible and the neuron-like element is useless. A very similar positive feedback process occurs if the initial chance reinforcement is negative. In this case the synapses become smaller and smaller (more negative or less positive) to no useful purpose. The neuronal elements are generally unstable in that small fluctuations in their reinforcement are soon turned into large ones without any particular relation to environmental reinforcement dependencies. (Sutton, 1978b)

3.2.3 Zerosetting

The early instability problems with the original reinforcement functions, based on what would correspond to membrane potential measured from resting, were present even with the use of a zerosetting mechanism. Later stability difficulties were partly due to the abandonment of any zerosetting mechanism because of a number of properties that have been seen as problematic for some models:

1. Most neurons probably fire more often than every 400 ms. These would be very rarely eligible with a zerosetting mechanism.
2. Zerosetting prevents learning at short delays. How can this be consistent with the fact that instrumental conditioning works dramatically better for shorter delays between response and reinforcement, even to delays less than 400 ms (Grice, 1948)?
3. Additional assumptions about network properties are necessary to explain even simple things like delay classical conditioning (Sutton, 1978b).
4. There is no possibility for learning about

temporally short feedback loops through the rest of the neural network.

3.2.4 Change in Input as Reinforcement

The most important change in Klopf's heterostat made in this early period was the switch from the use of input to the element as reinforcement to change in input as reinforcement. The proposal here was to make synaptic facilitation depend not on the absolute level of depolarization but rather on the amount of increase of depolarization following firing. If an output pulse is followed by an increase in depolarization, those excitatory synapses that were active when the output pulse was produced are facilitated, and those inhibitory synapses which were active are weakened. There were several reasons why this was a particularly interesting proposal. First, it became possible to eliminate zerosetting. The rationale for introducing zerosetting was to prevent a continuing high level of depolarization, caused by continued exposure to a given stimulus, from causing excessive adaptation. A model sensitive to stimulus change would be reinforced only when such exposure was initiated or terminated. Constant high levels of depolarization (or hyperpolarization) will not cause weight changes. Learning in animals often seems to

depend on the change in reinforcement rather than its absolute value. For example, the termination of punishment following a response will usually cause an increase in the frequency of that response.

A third line of thought leading to the use of the change in input as reinforcement was that of producing stability via an element-local negative feedback loop (as mentioned above). The idea was that synapses with larger weights must have their presynaptic signals followed by a proportionately larger input just to maintain their large weights. This introduced a natural limit on the growth of the weights that was dependent on the amount of reinforcement they indicated. One way of viewing this learning process is to regard each presynaptic signal as generating a prediction of how much input will follow, and then changing its weight according to whether that prediction was too large or too small. Some early models did this with each synapse making its own separate prediction of subsequent input (Sutton, 1978a). Another natural possibility was to add the predictions of synapses whose presynaptic signals occurred at about the same time to yield a composite prediction of subsequent input. It did not take long to recognize that this allowed a significant simplification: If predictions were proportional to synaptic strength, and they were added together to yield a

composite prediction, then simply the current total input was a prediction for later total input. Comparing the predicted with actual later input thus amounts to comparing past input with current input, i.e., to using the change in input as reinforcement. Section 4 contains one hypothesis as to how this might be done physiologically.

Viewing the use of change in input as reinforcement as involving a combination of predictions or expectations has turned out to be very useful in understanding it and relating it to animal learning theory. Sutton (1978c) used this approach to compare the behavior of such an adaptive element with a range of expectation phenomena in both classical and instrumental conditioning. In classical conditioning, the element was found to be closely related to a major descriptive model in animal learning theory due to Rescorla and Wagner (1972) which accounts for a broad range of expectation phenomena known as stimulus context effects. In instrumental conditioning, it was argued that the use of change in input as reinforcement was essential in explaining the full range of conditions under which learning takes place, including those in which no external reinforcer occurs. Figure 3.3 illustrates the range of possibilities involved.

The heterostat as originally proposed by Klopf, or any

REINFORCEMENT			
	HIGH	BASE	LOW
HIGH	NONE DURING OMISSION	- OMISSION	-- COMBINED EFFECTS
BASE	+	NONE NORMAL AFFAIRS	- PUNISHMENT CONDITIONING
LOW	++ COMBINED EFFECTS	+	NONE DURING AVOIDANCE ESCAPE, AVOIDANCE

Figure 3.3. Direction of learning changes (effective reinforcement) and name or description of the experimental paradigm corresponding to the nine basic cases of combinations of expectation and reinforcement. An element using change in input as reinforcement seemed necessary to begin to explain these phenomena.

heterostat using simple input levels as reinforcement, cannot explain these expectation phenomena as properties of the single element. Assumptions about network structure must be invoked in order to explain them. In addition, we have found that using the change in input as reinforcement does indeed alleviate the open-loop instability problems and the need for zerosetting. We have gradually become convinced that this is a genuine and important improvement. Most of our later models have taken the expectation based change in input element as a starting point. In the following we discuss it further.

3.2.5 The \dot{y} Element

This was the first element that we examined carefully that used the change in the total input as reinforcement rather than the absolute level of input. The term y refers to this according to a notational convention discussed below.

$$\Delta w_i(t) = w_i(t+1) - w_i(t) = c[y(t) - \bar{y}(t)]E_i(t) = c\dot{y}E_i \quad (3.5)$$

Here the eligibility E_i contains only the inverted-U shaped component and not the zerosetting component of the eligibility used in the weighted correlation model specified

above. The bar notation is used to denote the convolution of the barred time function with an exponential decay function. The easiest way to think of this is to regard each event in the barred function \bar{f} as causing a corresponding increment or decrement in \bar{f} , which then gradually fades or decays away with time. Figure 3.4 contains several examples which should make this clear. We will make frequent use of this decaying memory for producing eligibility traces. \bar{f} is usually assumed to be normalized such that if $f(t)$ is held constant at a particular value, then $\bar{f}(t)$ will asymptotically approach that constant value. This allows us to use $\dot{f}(t) = f(t) - \bar{f}(t)$ as a measure of f 's deviation from its recent past values, a measure closely related to the first time derivative of f . Many variations on this \dot{y} element have been considered, and the most interesting of these are discussed below.

3.3 Other Elements using Change in Input as Reinforcement

3.3.1 The Exponential Trace Eligibility \dot{y} Element

Because of the nature of the heterostatic theory of classical and instrumental conditioning, the eligibility computation is crucially involved in determining predictions for the effectiveness of learning as a function of the

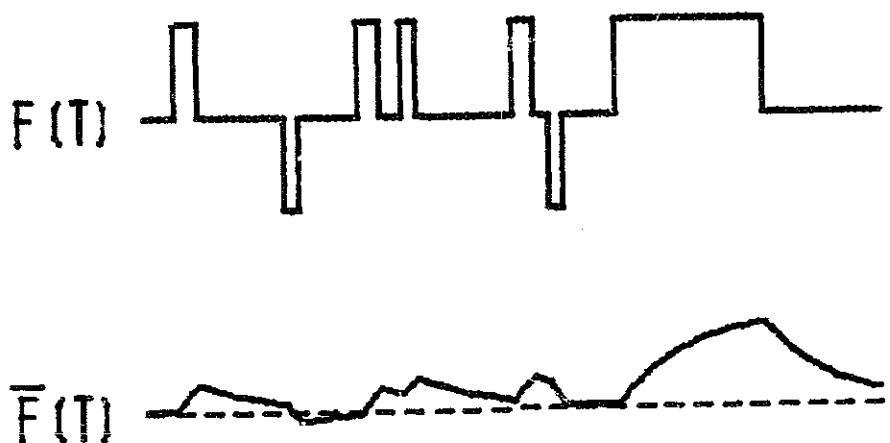


Figure 3.4. Illustration of the relationship between a function and the "bar" of that function. The bar indicates convolution with an exponential.

conditioned stimulus - unconditioned stimulus interval in classical conditioning, and of the conditioned response - reinforcement interval in instrumental conditioning. In the theory of animal learning, these two intervals are noted to have apparently similar effects on learning (which provides support for the heterostatic theory). However, the effect of this interval is not the same in all respects in classical and instrumental conditioning.

All the data on the effect of the conditioned response - reinforcement interval on learning in instrumental conditioning indicate better and faster learning the shorter the interval. The most careful studies have also shown reinforcement becoming essentially ineffective at intervals over five seconds (Grice, 1948). Thus, the plot of rate of learning versus this interval looks like Figure 3.5. An inverted-U shaped rate of learning versus interstimulus interval curve for classical conditioning is obtained by convolving this kernel with a conditioned stimulus signal of intermediate duration (Sutton, 1979). Thus, this sort of eligibility allows learning immediately after firing yet maintains the inverted-U shaped learning curve for classical conditioning. In April 1979 we converted our models to the use of an eligibility kernel function $f(k)$ of the monotonically decreasing shape of Figure 3.5.

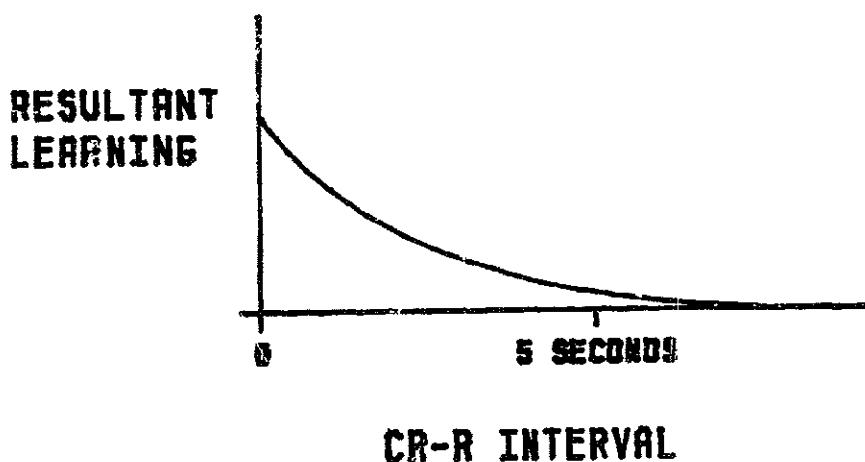


Figure 3.5. Resultant learning versus conditioned response-reinforcement interval in instrumental conditioning. This is also the form that instrumental conditioning indicates the eligibility increment function (kernel) should take.

The equation for the variation of the weights of the y element which uses an exponentially decreasing eligibility kernel is:

$$\Delta w \sim \dot{y} \bar{xy} \quad (3.6)$$

In words, the change in each synaptic efficacy or weight is proportional to the change in activity of the postsynaptic neuron times an exponentially decreasing weighted average of recent values of the product of pre- and post-synaptic activities.

3.3.2 The s Element

We consider this variation to be our current best heterostat, and we discuss this model further elsewhere.

$$\Delta w \sim \dot{s} \bar{xy} \quad (3.7)$$

where

$$s(t) = \sum_{i=1}^n w_i(t)x_i(t) \quad (3.8)$$

and

$$y(t) = \begin{cases} 1 & \text{if } s(t) + \text{noise}(t) > 0 \\ 0 & \text{else} \end{cases} \quad (3.9)$$

The replacement of \dot{y} by the closely related function \dot{s} has the advantage of allowing variation in the selection of action (the noise in the y computation) without introducing this spurious variation into the reinforcement term \dot{s} . Random variations in output do not act as reinforcement, as they do in the \dot{y} model, but are instead trials or experiments by the element with being both on and off. This learning equation can be given a natural physiological interpretation if s is thought of as the membrane potential of a neuron with a noisy threshold.

3.3.3 The "Dual" Heterostat

$$\Delta w \sim \dot{x} \cdot \overline{yx} \quad (3.10)$$

This rule can be arrived at by replacing each input variable (each x) in the \dot{y} element (Equation 3.6) by the output variable (y) and vice versa. Interestingly, the element retains the ability to do goal-seeking or instrumental conditioning-type learning. Conceptually, the element works like this: Each time the presynaptic neuron fires, it records which neurons it causes to fire (the \overline{xy} eligibility term). If it receives subsequent positive effective reinforcement - an increase in activity - it

concludes that it was "good" to make those neurons fire, and strengthens the synapses to them (if the subsequent effective reinforcement had been negative, the synapses would have been decreased). However, this rule does not produce classical conditioning-type learning as a side effect, although the normal heterostat does.

3.3.4 The Classical Conditioning Predictor Element

$$\Delta w \sim \dot{y} \cdot \bar{x} \quad (3.11)$$

In words: Presynaptic activity is correlated with subsequent changes in postsynaptic activity to determine the synaptic efficacy. This interesting element turns out to be a fairly good model of classical conditioning behavior as observed experimentally (see Section 4; Barto and Sutton, 1980). Important features are the use of an eligibility term, which allows genuine predictive learning in which the conditioned response can begin before the occurrence of the unconditioned stimulus, and expectation phenomena consistent with, and even going slightly beyond, current psychological theories of animal behavior. Since y is not present in the eligibility term, this element is insensitive to the effect of its past actions on current input, and thus is an

open-loop element. For this reason it is not capable of instrumental learning or other truly goal-seeking behavior. We did, however, gain a fairly complete understanding of its behavior and relationship to other theories (as presented in Section 4).

3.3.5 Dotting the x Eligibility Term

$$\Delta w \sim \dot{y} \overline{xy} \quad (3.12)$$

In classical conditioning experiments with animals, it is found that a crucial temporal variable determining ease of conditioning is the time interval between the onsets of the conditioned and unconditioned stimuli (the inter-stimulus interval). The \dot{y} term captures the dependency on the onset of the unconditioned stimulus (only at onset will this derivative measure be positive). Dotting the x term is meant to capture the dependency on the onset of the conditioned stimulus (CS) in the same way. In this element, a synapse becomes eligible only if it produces an output soon after presynaptic stimulation increases. If presynaptic stimulation decreases and an output is generated, then the synapse becomes negatively eligible - an increase in y will then decrease (and a decrease in y will

increase) the synaptic efficacy.

We will now argue that it is always inappropriate to use \dot{x} in triggering eligibility this way when it is x itself which is used with the weight vector to produce y . The argument, though simple, does require a detailed understanding of these adaptive elements. A synapse should be eligible if its presynaptic activity x was appropriately timed for influencing the observed postsynaptic firings y that may have caused the current reinforcement, \dot{y} . In this way, the synapses made eligible will be the ones which could have caused, or prevented, that activity, and which are thus responsible for the current reinforcement. The term \dot{x} , however, does not really meet this requirement. The simplest case is when x is maintained at a constant positive value for a long period of time. Throughout this time this x signal is influencing y and determining reinforcement, yet if \dot{x} determines eligibility, then this pathway would be eligible only at the start of the time period.

Our conclusion is that eligibility should depend directly on the variable used to calculate y from the weights. We prefer to always let x denote this variable. To make an element more sensitive to changes in stimulation levels, rather than absolute levels of stimulation, one can introduce a level of preprocessing on the input signals to

produce x signals whose absolute levels indicate changes in the original input variables (or whatever is thought to be most important in the original input signals).

3.3.6 Dotting the y Eligibility Term

$$\Delta w \sim \dot{y} \overline{xy} \quad (3.13)$$

or, combining with the previous rule,

$$\Delta w \sim \dot{y} \overline{\dot{x}y} \quad (3.14)$$

In these variations, changes in y , rather than y itself, are used to trigger eligibility. This possibility for eligibility is of particular importance for the specialized reinforcement models discussed below, and we have not yet analyzed it to our satisfaction. The most useful general observation seems to be that one's choice of either y or \dot{y} to trigger eligibility will depend upon which of these two most directly influences the change in reinforcement in the environments under consideration. For some environments, in which changes in reward are dependent on changes in output level, a \dot{y} term may be appropriate. In other environments, however, changes in reward may be due

directly to output levels rather than their changes (as in the landmark learning example in Section 6).

3.3.7 Separate x and y Averages in Eligibility

$$\Delta w \sim s \cdot \bar{y} \bar{x} \quad (3.15)$$

The most interesting aspect of this rule is that it can be thought of in two conceptually different ways. In the traditional approach, s is effective reinforcement and $y_b \cdot x_b$ determines which synapses are eligible. Alternatively, $\bar{s} \bar{y}$ can be thought of as effective reinforcement with just \bar{x} as eligibility. If $\bar{s} \bar{y}$ is positive, the neuron has had high output lately (\bar{y}) and this has been followed by an increase in input stimulation s . This suggests that the neuron is currently in a positive feedback loop from activity to stimulation, just the sort of situation in which we would like the neuron to fire strongly. Probably the equations using $\bar{x} \cdot \bar{y}$ eligibility can also be thought of in this way to some degree. Nevertheless, it seems more appropriate for eligibility to be positive only if positive postsynaptic activity y occurs during presynaptic activity x , because only in these cases could the synaptic weight influence the positive postsynaptic activity. For this reason, $\bar{x} \cdot \bar{y}$ is to

be preferred to $\bar{x} \cdot \bar{y}$.

3.3.8 Problems with using Change in Input as Reinforcement

Here we discuss some of the basic problems which have arisen with models that use the change in input as reinforcement.

3.3.8.1 The End Reinforcement Problem - By "the end reinforcement problem" we refer to the complex of problems that arises as one introduces "ultimate" or "primary" reinforcers into adaptive element models. According to Klopf's concept of generalized reinforcement, any input can become a reinforcer after it has occurred in appropriate relation to other inputs that are already reinforcers. It seems natural to define a few reinforcers as primary, and let others be built upon them. The idea is that in order for a signal to become a non-primary, or secondary reinforcer, it must occur in an appropriate relationship either to a primary reinforcer or to a secondary reinforcer, which in turn must occur in an appropriate relationship to a primary reinforcer (or to another secondary reinforcer, which must in turn...). Thus, the primary reinforcers are in some sense the ultimate justification of all secondary

reinforcers. They are often thought of as occurring at the end of a sequence of secondary reinforcers, and this is the rationale for the term "end reinforcers." The end reinforcement problem is either to do away with the apparent need for these special reinforcing inputs, or else to use them in some manner which avoids the difficulties discussed below.

As an example, let us consider the end reinforcement problem for the classical conditioning predictor element briefly discussed above (Section 3.3.4). For a number of reasons it is desirable that a non-primary excitatory input signal that is not followed by reinforcement should result in a decrease in the associated synaptic weight. In the classical conditioning element, this occurs due to the large decrease in input at the offset of this excitatory signal. If this excitatory signal is a primary or end reinforcer, however, then we do not want it to decrease, even if not followed by another reinforcer. A simple solution, and the one used in our published results, is to specialize the end reinforcers to the extent that their synaptic weights are unaffected by the learning process - they arrive over pathways having fixed weights.

However, there is reason to suspect that this sort of crude solution will never be completely adequate, either for

this particular element or for generalized reinforcement models using expectation in general (including the s element). Activity in the fixed end reinforcer pathway will cause an unusual level of activity and thus a subsequent unusual level of expectation. Yet, the pathway's strength is immune from the results of this expectation, which would otherwise drive the connection weight toward zero. Other pathways that may be active as the end reinforcer occurs are not so immune however, and this is where the hidden problem arises. It seems to be a reasonable assumption that other purely informative signals should not seriously impair the element's behavior. Yet, consider what happens if we assume that a signal is available to the destination element over one of its variable pathways which is the same signal (in terms of its time course) as one which arrives via the fixed pathway. This variable pathway will have an eligibility identical to that of the fixed pathway, but it is not immune to the lack of reinforcement. When their common signal occurs, expectation is built up or maintained in the destination element. When the signal ends, there is expectation without activity. The fixed association is immune, but the variable association is driven away from zero in the direction opposite to that determined by the sign of the fixed pathway: If the fixed connection is excitatory, the plastic connection will become inhibitory; if the fixed is inhibitory, the plastic will become

excitatory. The next time the signal occurs, there will be a reduced effect on the destination element because the influence of the fixed pathway will be slightly counteracted by the influence of the variable pathway. The variable weight will continue to change as before. This process continues until the effects of the fixed and variable pathways exactly counterbalance. The end reinforcer signal, in spite of all our ad hoc efforts to make it fixed and non-zero, will produce no net effect on the element.

End reinforcers and the secondary reinforcers they support seem to be fundamentally different. Secondary reinforcers must always generate a prediction or expectation of another reinforcer soon to come, whereas an end reinforcer should not. This suggests that the input lines to an adaptive element that are designated the end reinforcers must play a special role in the learning equation. This is not, however, a return to specialized reinforcement models, for although there would be designated specialized end reinforcer signals, any signal can take on reinforcing properties by association with these end reinforcers.

3.3.8.2 Conflict between the Selecting and Reinforcing Functions of Input - As Klopf has noted in his reports, when one uses input to an adaptive element, or change in that input, as reinforcement, there is occasionally a conflict between the informative, or selecting, and the reinforcing functions of input. The type of conflict between selecting and reinforcing functions that concerns us can be seen in the following example. If an element should not be active, and gets rewarded (excitation or increase in activation) for being inactive, then the natural effect of that reward will be to make the element become active. Only with difficulty and long training could the element learn to remain inactive. Some of our simulation experiments have provided evidence that this sort of conflict does indeed exist. On some simple goal-seeking tasks (the associative search problem, discussed in Section 5) our best heterostat has been observed to perform less well, or at least less robustly, than elements whose selective and reinforcing functions were separated into separate input lines. This evidence is hardly conclusive, for the task was not one that would demonstrate the special abilities and advantages of the heterostat, and indeed these might be expected to get in the way. However, this sort of result has encouraged us to look at elements with specialized reinforcing input lines.

3.4 Models with Specialized Reinforcing Input Lines

In the preceding sections we have discussed adaptive elements which use their excitation and inhibition as reward and punishment. These are generalized reinforcement models. For comparison, as well as due to their own intrinsic interest, we have also developed and compared a number of specialized reinforcement models, i.e., models whose evaluation signal is provided via a unique pathway clearly separated from those which affect the activity of the element. These specialized reinforcement models have played an important role in the development of our current heterostat models. In the last model we will discuss, for example, excitation and inhibition do act in the usual way as reinforcers, but a specialized non-exciting input line also provides reinforcement. Such combinations may alleviate some of the problems with the heterostats we have seen so far.

In the following, let z denote the specialized reinforcement signal, also called the payoff (see Figure 4.1).

3.4.1 ALOPEX as an Action Selector

$$\Delta w(t) = c[z(t) - z(t-1)][y(t-1) - y(t-2)]$$

or

$$\Delta w \sim \dot{z} \cdot \overline{\dot{y}} \quad (3.16)$$

where

$$y(t) = \begin{cases} 1 & \text{if } w(t) + \text{noise}(t) > 0 \\ 0 & \text{otherwise} \end{cases} \quad (3.17)$$

z(t) a function of y(t - 1)

This element is simply a different way of viewing the components of the ALOPEX system of Harth and Tzanakou (1974). The learning equation does not use any input signal information other than the payoff signal z. The element only learns to set its action at that level which maximizes its payoff input z - it cannot vary this action level as a function of other input. The following element is the extension of this one to include a sensitivity to input information other than the payoff signal.

3.4.2 Associative Search Network Element

$$\Delta w \sim \dot{z} \overline{x \dot{y}} \quad (3.18)$$

z(t) a function of y(t - 1) and x(t - 1)

In addition to the payoff signal, this element is sensitive to other input information. We think of the signals on these other pathways as indicating some situation in which the element is to act. The element takes note of the situation in which an increase in y increases z , and then, by changing the appropriate weights, only increases the y level in those situations. This element is related to some learning automata considered by Tsetlin and his followers (Tsetlin, 1973), although learning automata are always taken to have only the payoff input, z . This element is discussed extensively in the next section.

3.4.3 The Associative Search Problem

The associative search problem (Section 5) is a task rather than a particular learning equation. An associative search net is defined in contrast to the associative memory systems that are usually discussed in the literature. In such a "standard" associative memory, input patterns, or keys, and desired recollection patterns are presented simultaneously during training. After this training phase the associative memory should produce the recollection when given the key. An associative search net (ASN) also should produce a particular output pattern for each key, but the ASN is never provided with that desired recollection during the training phase. During the training of an ASN, the

environment provides key input, waits for an output or action from the ASN, and then provides a scalar evaluation of that action as a function of the key provided. For each key input pattern, the ASN must search for the action which maximizes its reward in that input situation. A bank of associative search net elements (Figure 3.6) can solve this sort of problem under certain conditions. This network is discussed more fully in Section 5.

3.4.3.1 Nulled Transitions - There is a problem with using the system shown in Figure 3.6. During the transition from one input situation to another, there may be a large change in z . The system mistakenly thinks it was its last action in the first situation that caused the transition to the second. Since we want the system to produce the best action in each situation, we would like this change to be ignored. The simplest way to do this is to prevent any learning from occurring during the transitions between situations. There are at least two very different objections to preventing learning during transitions between input situations. The first is that we can do better than this by adding a predictor, and the second suggests that we really want the system to try to control its transitions from situation to situation, and that the changes in z at the transitions should not be ignored at all. These issues are discussed more fully in Section 5.4.

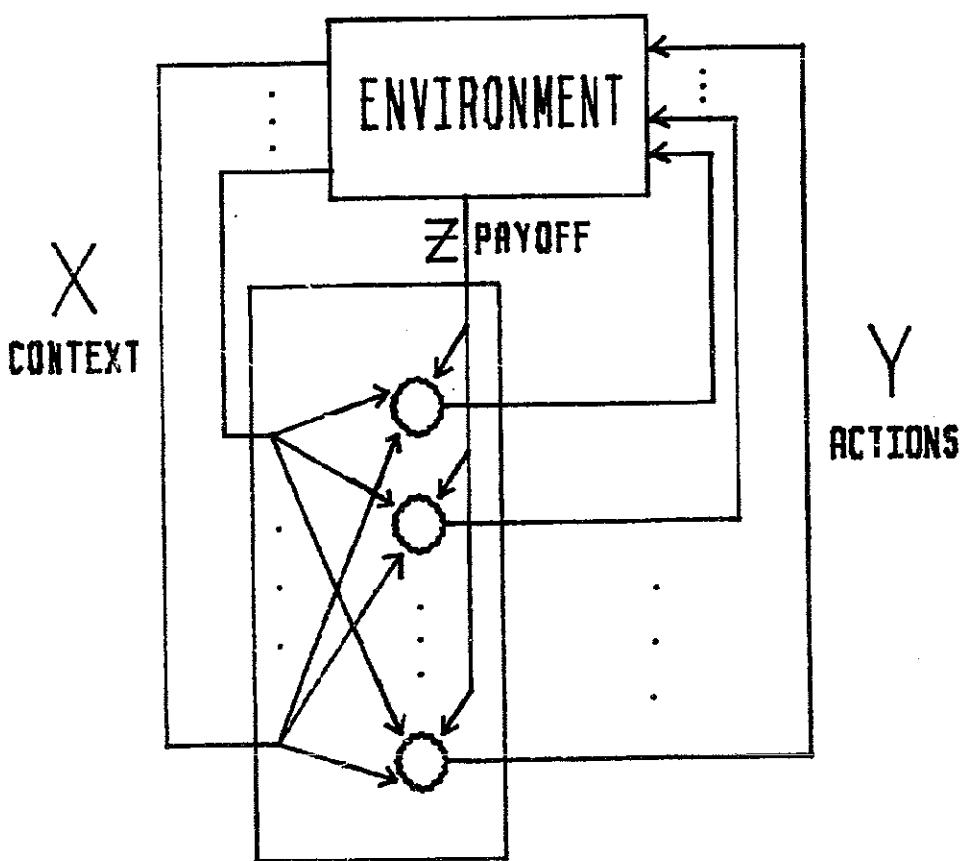


Figure 3.6. A simple associative search net made from associative search elements.

3.4.3.2 Associative Search with Predictor Element -

$$\Delta w \sim (z - \bar{p}) \cdot \bar{x} \dot{y} \quad (3.19)$$

$$p(t) = \sum_{i=1}^n w_i(t)x_i(t) \quad (3.20)$$

$$\Delta w_p \sim (z - \bar{p})\bar{x} \quad (3.21)$$

See Figure 3.7

This element uses the deviation of reward from predicted or expected reward to change weight, rather than the deviation of current reward from the reward at the last time step. Thus, when a new input situation is encountered, a predicted level of reward for the new situation becomes immediately available, and it is not necessary to prevent learning from occurring during the transition. This rule would work even if the input situation changed every time step, whereas a nulled transition system would not work at all if this were the case.

It is important to note that the prediction used here is a non-anticipatory prediction. In other words, the

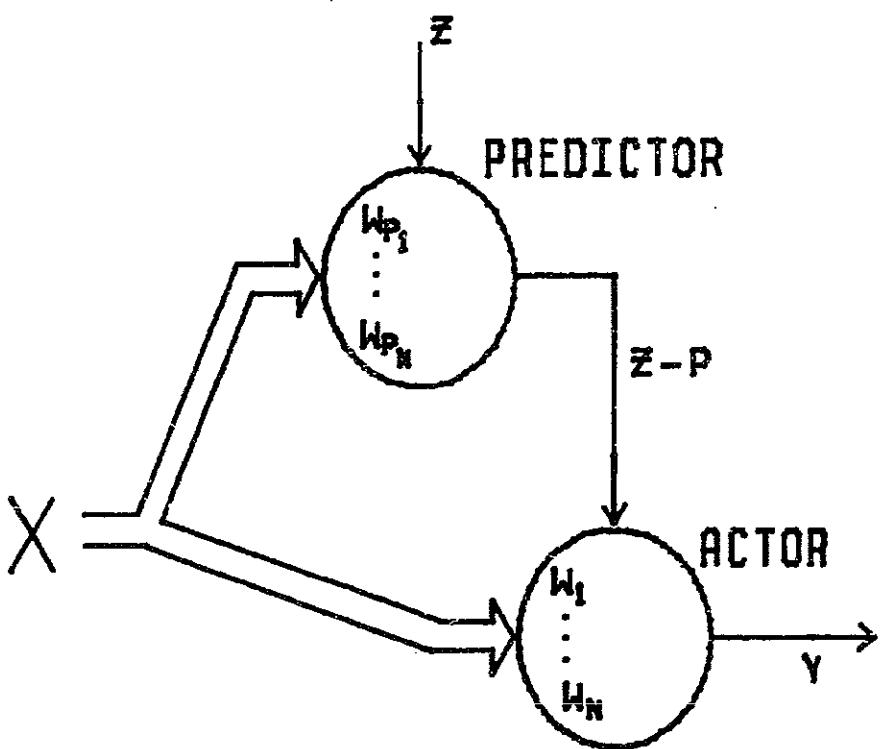


Figure 3.7. Diagram of the associative search net with predictor element. This element consists of a predictor part and an actor part, each with its own set of weights.

prediction is not used until the event predicted has already occurred and been observed. This is still useful because it allows the detection of deviations of reward from the predicted level. Later we will discuss elements that combine this with a genuinely anticipatory prediction (a prediction used before the predicted event is available).

3.4.3.3 Learning Situation Transitions - If the reward that can currently be attained differs from situation to situation, then it is reasonable to expect an adaptive system to attempt to control which situation it is in, i.e., to control its environment in order to cause it to present a situation in which a high reward can be attained. Of course, the adaptive system may not be able to affect which situation is presented next (this is the case in the associative search task), but why not expect it to try? If it were trying, the changes in z at situation transitions would be very important informative cues, and should not be ignored at all.

Below we consider several elements designed to solve the problem of maintaining a high payoff, both by choosing high payoff actions within a situation, and by choosing actions that control the environment in order to cause the

occurrence of situations in which higher payoffs are available. The elements differ only in their reinforcement terms. In all cases eligibility will be the standard $\bar{x} \cdot y$ discussed above.

The associative search net element uses \dot{z} as a reinforcement term. This rule evaluates past actions in the situation that generated them (represented by the $\bar{x} \cdot y$ eligibility) according to the change in payoff that resulted, ignoring the situation altogether for evaluation purposes. The first objection that one might have to this element is that it does not use the situation input vector to give it some idea of how much payoff is possible in each situation. For example, assume that in a certain situation X_0 the payoff drops significantly whether the element chooses either a high or a low activity. With continued experience with X_0 the element should learn to make the best of a bad situation and choose the action which results in the smaller decrease in payoff. However, whatever action is chosen, it will be punished according to the learning rule implemented by the associative search net element. The better action will be made less likely to occur every time that it does occur. (A similar problem occurs if there is a situation X_1 in which there is always an increase in payoff, whatever action is taken.) This problem suggests that the element could be improved by accumulating an expectation for

each situation of what the change in payoff will be. Then the actual change could be compared with this expectation.

This idea is similar to that behind the associative search net with predictor element. However, in that element, we constructed a prediction of upcoming z value, whereas we construct a prediction of upcoming changes in z value in the payoff change predictor:

$$\Delta w \sim (\dot{z} - \bar{q}) \bar{xy} \quad (3.22)$$

where $\bar{q}(t)$ is a prediction of change in payoff at t, based on recent situations.

$$q(t) = \sum_{i=1}^n w_{qi}(t) x_i(t) \quad (3.23)$$

$$\Delta w_q \sim (\dot{z} - \bar{q}) \bar{x} \quad (3.24)$$

Here synaptic efficacies are increased only if a greater increase in payoff is received than was expected. Unlike the associative search net element, this element can adjust to situations from which only increases or decreases in payoff are possible, and still learn effectively.. Another

nice feature of this element is that weights change only enough to make sure the right action is made every time. In the associative search net element, on the other hand, weights are changed every time payoff changes, even if the best action has already been found and the payoff change could have been completely anticipated.

3.4.4 An Element that Makes Two Uses of Prediction

$$\Delta w \sim [(z - \bar{p}) + \dot{p}] \overline{xy} \quad (3.25)$$

where

$$\Delta w_p \sim (z - \bar{p}) \overline{x} \quad (3.26)$$

This element is best seen as an extension of the predictor based element discussed above (Equations 3.19, 3.20, and 3.21) to solve the associative search problem. That element used only $z - \bar{p}$ as effective reinforcement, whereas this rule adds a \dot{p} term. That element, like the payoff change predictor, does not use the new situation input to evaluate action. This can be important, since often the situation input can be an indicator of what opportunities for payoff lie ahead. The second term, \dot{p} , in this element's reinforcement term makes use of this

situational information about upcoming payoff. Since p is a predictor of future payoff, the change in p can be considered as rewarding as a change in payoff itself. In this element, change in prediction of reward is merely added into the reinforcement term of the payoff change predictor. By a similar reasoning process, we might expect that an element that uses $\dot{z} + p$ as effective reinforcement might work well. However, at this point our understanding has progressed sufficiently that we can attempt a more theoretical presentation.

3.4.5 A Proposal for an Alternative Problem

The primary purpose of considering the associative search problem in our research has been to serve as a focal point in the evaluation of various learning rules. As our learning rules and our understanding of them has evolved, so has our understanding of the problems we would like them to solve. Most problems ignore certain issues in order to focus on others, and the associative search problem is no exception. The associative search problem directs our attention to a simple, stark form of situation sensitive search for optimal actions. We would like now to propose a problem which retains the emphasis on situation sensitive goal-seeking, but which also introduces two additional considerations. In this new problem, unlike the associative

search problem, the actions of the adaptive system will be allowed to have an effect on which situation it next finds itself in. Thus, we will want the adaptive system to learn to control its environment to cause the occurrence of those situations in which a high payoff can be attained. Second, this ability to control the environment raises a major new complication: There may be times when the highest payoff can only be reached by passing through a temporary period of low payoff. Similarly, an action which brings immediate high payoff may inevitably be followed by a prolonged period of very low payoff. In these cases, some sort of evaluation extending over many time steps would seem to be essential for successful adaptation. Thus, a new formulation of optimal and adaptive behavior is required.

Several remarks are in order about some aspects of adaptation that this new formulation is not focusing on. The following are three assumptions made in the new formulation which keep the problem reasonably limited. Of course, as success is achieved on this delimited problem, these assumptions can be gradually weakened or removed.

1. The identification of the states of the environment is assumed to be simple given the situation information. It is sufficient to assume each environmental state activates a unique situation

input line.

2. The next environmental state is assumed to be dependent only on current state and current action. In other words, all influences through the environment have a delay of exactly one time step.
3. The element will not be expected to improve its input representation with experience in any way.

3.4.5.1 The Problem Schema -

$$Q(t+1) = \text{ENV}\{Q(t), y(t)\} \quad (3.27)$$

$$y(t) = \langle W(t), X(t) \rangle = \sum_{i=1}^n w_i(t)x_i(t) \quad (3.28)$$

$$X(t) = \Phi\{Q(t)\} \quad (3.29)$$

$$z(t) = Z\{Q(t)\} \quad (3.30)$$

$$W(t) = f\{W(t-1), z(t), X(t-1), y(t-1)\} \quad (3.31)$$

See Figure 3.8.

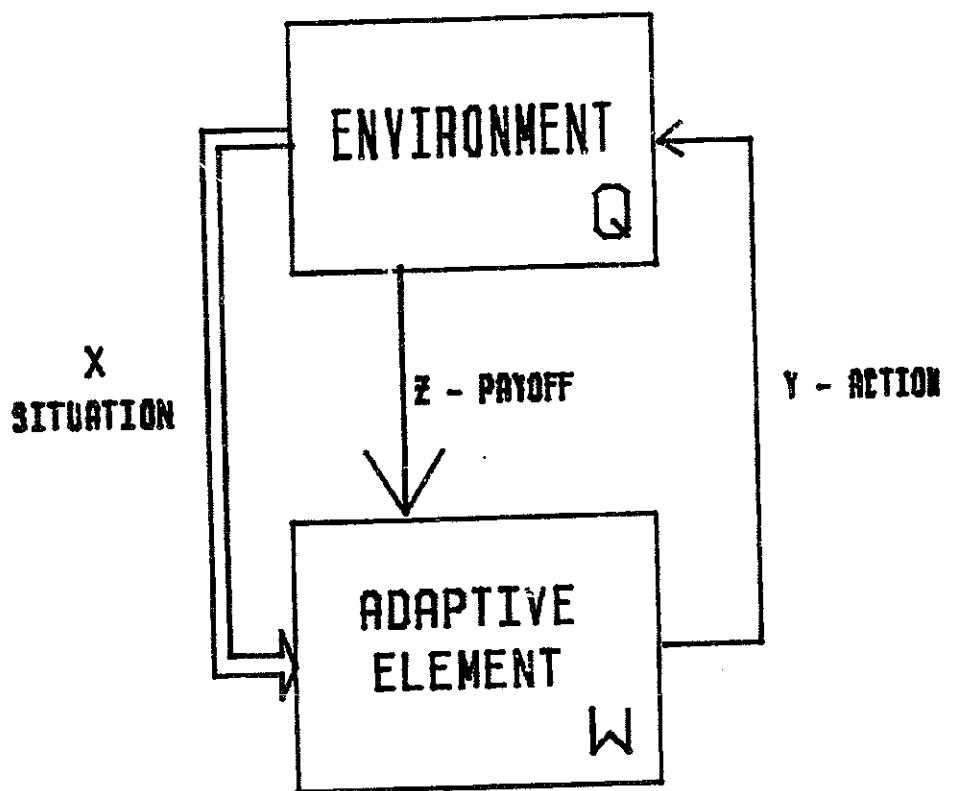


Figure 3.8. The element task schema.

3.4.5.2 Payoff Functions - In the above specification, $z(t)$ is what is ordinarily thought of as the payoff function. In fact, however, this momentary measure can be a very poor measure of the correctness of a certain action. $z(t)$ is the instantaneous payoff, but an action that results in immediate reward may result in low payoff later. As discussed earlier, it is necessary to introduce a new notion of optimality to begin to deal with these delayed effects.

For a particular adaptive system and environment one can define a function of time called the ideal payoff function $U(t)$ which gives a measure of how well that adaptive system has done, taking into account the consequences of past actions which have not yet materialized. We take as one such measure simply the sum of the z values, both already received and yet to be received, the latter in general will only be partly determined:

$$U(t) = \sum_{T=0}^t z(T) + E\left[\sum_{T=t+1}^{\infty} z(T) | Q_A(t), Q_E(t)\right] \quad (3.32)$$

where $E\{ \cdot \}$ indicates the expected value operator and $Q_A(t)$ and $Q_E(t)$ denote the states at time t of the adaptive system and the environment respectively. A nearly equivalent definition of the ideal payoff that is sometimes more useful is to define it to be the expected average level of $z(t)$ over the lifetime of the organism:

$$U(t) = \lim_{\tau \rightarrow \infty} \frac{\sum_{T=0}^t z(T) + E\{\sum_{T=t+1}^{\infty} z(T) | Q_A(t), Q_E(t)\}}{\tau} \quad (3.33)$$

Our approach to situation sensitive goal-seeking is to think of the element as varying its action in order to receive some feedback as to the evaluative effect of the variation. Ideally, the effect on the ideal payoff would be available, and the weight change equation would be simply

$$\Delta w \sim \Delta U \cdot \bar{xy} \quad (3.34)$$

Of course, ΔU is no more available to most adaptive systems than U is, and our adaptive elements will have to settle for some approximation to it:

$$\Delta w \sim \hat{\Delta U} \cdot \bar{xy} \quad (3.35)$$

where $\hat{\Delta U}$ is the adaptive element's estimation of the change in U .

The following is an example of how this approach might be used. First, note that U as defined in Equation 3.33

consists of two parts, one which can be completely known by the adaptive system and one which generally can not:

$$U(t) = \sum_{T=0}^t z(T) + E\left(\sum_{T=t+1}^{\infty} z(T) | Q_A(t), Q_E(t)\right) \quad (3.36)$$

This suggests taking as the estimate \hat{U} of U merely the known part:

$$\hat{U}(t) = \sum_{T=0}^t z(T) \quad (3.37)$$

To yield an adaptive element, we solve for the change in $\hat{U}(t)$

$$\Delta \hat{U}(t) = \hat{U}(t) - \hat{U}(t-1) = \sum_{T=0}^t z(T) - \sum_{T=0}^{t-1} z(T) = z(t) \quad (3.38)$$

Substituting this into Equation 3.35 immediately yields the associative search element. This is in fact a good way of understanding that element. The associative search element builds no estimate of what future payoff will be. From this follows the two flaws of that element. First, every positive z is seen as a reward and every negative one as a punishment. The z value is always seen as a deviation from zero rather than from some prediction, or estimate, of z . Second, with a zero estimate of z for the future, it cannot

recognize delayed effects of its actions on z . Thus, the element cannot use secondary reinforcement; that is, it cannot learn that a situation will soon be followed by reward and should be interpreted as reward itself.

This analysis clearly suggests that it may be useful to use an estimate \bar{U} -hat of U which includes some estimate of future values of z . One possibility currently under investigation is to form a prediction of z with a separate set of prediction weights, much as we did earlier for the associative search with predictor element. Here, however, we want the prediction to extend several time steps into the future. This suggests the following equation:

$$\bar{p}(t+1) = \bar{p}(t) - \alpha \bar{p}(t) + \alpha p(t) \quad (3.39)$$

$$0 < \alpha \leq 1$$

where

$$p(t) = \sum_{i=1}^n w_{pi}(t)x_i(t) \quad (3.40)$$

and

$$\Delta w_p \sim \Delta \bar{U} \bar{x} \quad (3.41)$$

where $\bar{p}(t)$ is the prediction of $z(t)$. As an estimate of z 's values in the future, we can use the values \bar{p} will take in the future, assuming no more input events occur, or that

their net effect will be zero. This lets us use Equation 3.32 for U to form a new estimate \hat{U} :

$$\hat{U}(t) = \sum_{T=0}^t z(T) + \sum_{T=t+1}^{\infty} \bar{p}(T) \quad (3.42)$$

$$= \sum_{T=0}^t z(T) + \sum_{T=t+1}^{\infty} \bar{p}(t+1)(1-\alpha)^{(T-t+1)}$$

$$= \sum_{T=0}^t z(T) + \frac{\bar{p}(t+1)}{\alpha}. \quad (3.43)$$

Taking the difference of both sides yields

$$\begin{aligned} \Delta \hat{U}(t) &= z(t) + \frac{\bar{p}(t+1) - \bar{p}(t)}{\alpha} \\ &= z(t) + \frac{-\alpha \bar{p}(t) + \alpha p(t)}{\alpha} \\ &= z(t) - \bar{p}(t) + p(t) = z(t) + \dot{p}(t) \end{aligned} \quad (3.44)$$

This suggests, via our general Equation 3.35, the following adaptive element:

$$\Delta w \sim (z + \dot{p}) \bar{xy} \quad (3.45)$$

with

$$\Delta w_p \sim (z + \dot{p}) \bar{x} \quad (3.46)$$

This interesting element, and others of a similar nature, are currently under investigation.

SECTION 4

OPEN-LOOP LEARNING: EXPECTATION, PREDICTION AND CLASSICAL CONDITIONING *

4.1 Introduction

One way to bridge the gap between behavioral and neural views of learning is to postulate neural analogs of behavioral modification paradigms. Hebb's suggestion that when a cell A repeatedly and persistently takes part in firing cell B, then A's efficiency in firing B is increased, is the most familiar of these postulates (Hebb, 1949). This rule for synaptic plasticity is a neural analog of associative conditioning and continues to exert a powerful influence on theoretical and experimental research in learning and memory. Neural network models designed to

* This section is based on a paper entitled "Toward a Modern Theory of Adaptive Networks: Expectation and Prediction" by R. S. Sutton and A. G. Barto to be published in Psychological Review, 1981.

explore the behavioral possibilities of modifiable structures typically employ a pre- and postsynaptic correlation for altering connectivities as a mathematical representation of Hebb's postulate (e.g., Anderson, Silverstein, Ritz, and Jones, 1977; Brindley, 1969; Grossberg, 1974; Kohonen, 1977; Marr, 1969; von der Malsburg, 1973). However, in addition to the fact that there is no direct experimental support for the Hebbian rule as a model of neural plasticity, several different bodies of evidence have accumulated that suggest that such simple contiguity rules can account neither for the behavioral facts of learning, nor for the theoretical necessities of successful adaptation.

The analysis of elemental processes of learning has a long tradition within animal learning theory. To a large extent it has been successful: Fundamental laws of wide, if not complete, applicability have been found. Animal learning theory constitutes a large body of carefully explored and tested theories about fundamental processes of learning. Given this, it is surprising how little contact and interaction there have been between animal learning theory and adaptive systems theory, particularly insofar as the latter attempts to mimic neural networks or biological adaptive systems in general.

Numerous adaptive systems papers have made brief reference to basic animal learning processes such as classical and instrumental conditioning. But, almost exclusively, inadequate models of these conditioning processes have been used, and in some cases they are so inadequate that while a theorist derives support for his model by citing a learning process, in reality the experimental evidence and modern learning theory contradict even the simplest predictions of the model. Classical conditioning involves an interplay between expectations and stimulus patterns that is too complex to incorporate into a simple correlation rule such as Hebb's. The common modifications of a correlation rule, for example the introduction of delay in input or output pathways, result in behavior still not in agreement with experimental data. Moreover, as we argue below, the phenomena actually observed in classical conditioning is perhaps crucial for sophisticated adaptive behavior.

The history of attempts to construct adaptive networks of neuron-like components also suggests that something essential is not preserved by the Hebbian model and its variants. Network approaches to adaptive system design have been notable in their failure to produce learning behavior beyond a rather low level of sophistication. The information processing success of adaptive networks is

restricted almost entirely to moderate success in the recognition, processing, and associative storage and retrieval of spatial patterns. There is a conspicuous absence of nontrivial processing of temporal patterns. It may be true that in the brain some kinds of temporal patterns are processed by being represented spatially as, for example, suggested by Lashley (1951), and some models use this principle (e.g., Fukushima, 1973; Grossberg, 1969; Spinelli, 1970). However, little progress has been achieved in our understanding of how a system can both learn and effectively use knowledge while interacting in real time with a complex environment. Yet these temporal aspects of a system's interaction with its environment are central to much intelligent behavior.

In the time since the first computational experiments with adaptive networks were carried out, remarkable advances in the understanding of the cellular basis of behavior have occurred. In recent years, invertebrate animals have been successfully used to study aspects of the neural basis of behavioral modifications (e.g., Kandel, 1976, 1978). Although this approach has not yet elucidated the cellular basis of associative learning, simpler but possibly related forms of nonassociative learning have been successfully analyzed at the cellular level. These studies reveal that neurons employ a wide variety of biochemical modulatory

processes that interact in complex ways with electrical activity and that this interaction mediates forms of behavioral modification (Kandel, 1978).

Despite this evidence that neurons are capable of very complex information processing, adaptive network theorists continue to produce idealized neural element designs which are constrained by the early view that neurons are essentially switching elements having little internal processing power. Although one of the most important aspects of model building is simplification, the lack of significant progress in adaptive network theory, together with the high complexity of cellular and synaptic machinery, suggests that these idealizations leave out some mechanisms that are essential for producing sophisticated adaptive behavior.

In this section we describe an adaptive element model which is more reasonably in accord with the facts of modern animal learning theory than models commonly used in adaptive network research. After discussing several forms that adaptive element analogs of classical conditioning have taken in the past, we briefly introduce our model. We then present the basic elements of a view of classical conditioning that is more realistic than that commonly used in adaptive network studies. We show how the behavior of

our model is in good agreement with a variety of aspects of animal learning data. We then discuss how our model is related to a variety of other adaptive elements which form part of adaptive system theory. No attempt is made to be exhaustive. Learning theory is a complex subject with many controversies, and adaptive system theory is extremely diverse. We have tried to abstract from the very large animal learning theory literature those points on which there is a reasonable amount of agreement and which we consider to be most pertinent for adaptive network modelling and simulation.

Despite recent advances, it is still premature to propose a testable molecular model of associative learning. However, even though we see our model as being of interest primarily from behavioral and theoretical perspectives, we speculate as to how the cellular mechanisms which are beginning to be elucidated could implement the required computations. Our purpose in doing this is two-fold. First, we desire to demonstrate that processing of the proposed complexity is clearly possible at a cellular or simple network level. Second, some aspects of the proposed learning rule can be implemented so naturally by known mechanisms that a discussion of these mechanisms in light of our behavioral and theoretical observations, while speculative, may contribute to experimental efforts to

understand neuronal plasticity.

Although we restrict attention in this section to classical conditioning, our research was motivated by an interest in more complex forms of learning and, in particular, the novel suggestion by Klopff (1972, 1979, 1981) that neurons may be reinforcement learning devices of a kind fundamentally different from those previously proposed in neural theories. The aspects of classical conditioning which we consider here form a necessary prelude to moving beyond the restrictions of the classical conditioning paradigm.

Finally, although our theory is an attempt to explore the consequences of attributing quite complex computational power to individual adaptive elements, it is not our intention to suggest that all of the mechanisms must necessarily reside in each element. Rather, our program of endowing a single adaptive element with behavior having detailed properties of classical conditioning represents our feeling that these properties are fundamental to adaptive behavior. In particular, what we call an adaptive element may not correspond to a single neuron.

4.2 Adaptive Element Analogs of Classical Conditioning

In a simple classical conditioning experiment the subject is repeatedly presented with a neutral conditioned stimulus (CS), i.e., a stimulus that does not cause a response other than orienting responses [footnote], followed by an unconditioned stimulus (UCS) that reflexively causes an unconditioned response (UCR). After a number of such pairings of the CS and the UCS - UCR, the CS comes to elicit a response of its own, the conditioned response (CR), that closely resembles the UCR or some part of it. For example, a dog is repeatedly presented with first the sound of a bell (the CS), and then its food (the UCS) which causes the dog to salivate (the CR). This simplified description of classical conditioning leaves much unsaid, as we shall see.

In studies of the cellular basis of learning and in purely theoretical studies of adaptive systems it is frequently convenient to postulate neuron-like mechanisms which embody various types of "learning rules". The rules

Strictly speaking, this stimulus is not a conditioned stimulus until the animal has begun to be conditioned to it. However, as is often done, we simplify notation in this section by referring to any stimulus that is meant to be considered as eventually or potentially becoming conditioned as a conditioned stimulus.

describe how the strengths of interconnectivity change between units that are intended to be crude models of neurons. In keeping with this tradition, we shall sometimes refer to synapses, synaptic weights, etc., but the reader should remain mindful that the relationship between models of this form and neural plasticity is often one of coarse analogy. We prefer to think of the rules as describing the behavior of "adaptive elements".

Figure 4.1 shows an element with input signals x_1, \dots, x_n , connection weights w_1, \dots, w_n , output y , and a specialized "teacher" input z . Since we wish to focus only on rules for changing the weights w_i , we will not pay particular attention to the input-output function of the element. For our purposes, it suffices to say that y is some function of the weighted sum of the inputs; that is, for any time t ,

$$y(t) = f[\sum_{j=1}^n w_j(t)x_j(t)], \quad (4.1)$$

where f is a function which resembles the one shown in Figure 4.2 [footnote]. Of course, when an adaptive element is proposed as an analog of animal learning, the form of

According to Equation 4.1, the adaptive element computes its output y instantaneously from its inputs x_i . In order to remedy the problematic consequences of this when networks are considered, one can assume that a small delay exists in the communication links between the elements. For our present purposes, we do not need to consider this detail.

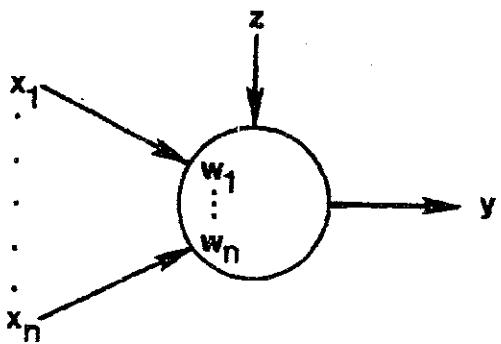


FIGURE 4.1. An adaptive element with n modifiable input pathways x_i , $i = 1, \dots, n$, connections weights w_i , $i = 1, \dots, n$, a specialized input z required by some adaptive elements to transmit the signals of a "teacher", and an output labeled y .

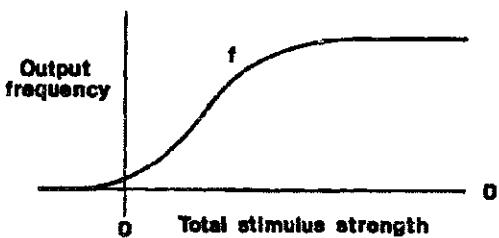


FIGURE 4.2. A common form of nonlinear input-output function used in neural and adaptive element models. When these models are used in analogs of conditioning experiments, this function becomes a response mapping rule.

this function becomes crucial in making precise predictions about behavioral data. In these cases, the function is related to response mapping rules (e.g., Frey and Sears, 1978). If the adaptive element is proposed as a neuron model, this function relates, for example, the firing frequency of a neuron to its membrane potential.

For an adaptive element analog of conditioning, the presence of CS_i , $i = 1, \dots, n$, is indicated by activity on the corresponding input pathway x_i . For example, if $x_i(t)$ denotes the signal on pathway x_i at time t , then the presence of CS_i at time t can be indicated by letting $x_i(t) = 1$. If CS_i is not present, $x_i(t) = 0$. The associative strength of each CS_i at time t is $w_i(t)$, the weight associated with pathway x_i . The CR is identified with the output y so that by Equation 4.1 the associative strengths of the CS_i , $i = 1, \dots, n$, determine the magnitude of the CR. Learning rules take the form of equations for changing the values of the weights w_i , $i = 1, \dots, n$, over time as functions of various aspects of the element's inputs and outputs. Usually the element's behavior is intended only to qualitatively resemble animal learning data.

The most well known example of an adaptive element analog of classical conditioning is based on Hebb's neural postulate that persistent pairing of pre- and postsynaptic

activity increases a pathway's efficacy (Hebb, 1949). Although Hebb did not provide a mathematical formulation of this rule, the following expression has been widely used to implement his postulate:

$$w_i(t + 1) = w_i(t) + c x_i(t)y(t) \quad (4.2)$$

where c is a positive constant determining the rate of learning. Here, and throughout this section, we use a time step of one unidentified unit that can be set equal to various values to suit particular interpretations of a model. For the case in which the input signals x_i and the output signal y are binary valued, w_i is incremented by c whenever an input pulse arrives and the cell fires and is unchanged otherwise. For the case of real valued signals, w_i becomes a rough measure of the correlation between input signal x_i and output signal y . Unlike several other rules, this rule does not require the specialized "teacher" input shown in Figure 4.1.

It is easy to see how a Hebbian learning rule can implement a simultaneous contiguity view of classical conditioning (Figure 4.3). Suppose a Hebbian adaptive element has an excitatory UCS input pathway having weight w_{UCS} sufficiently large so that UCS occurrence causes the element to respond with the UCR. If the element also has an input pathway for the CS having an initially low weight w_{CS} ,

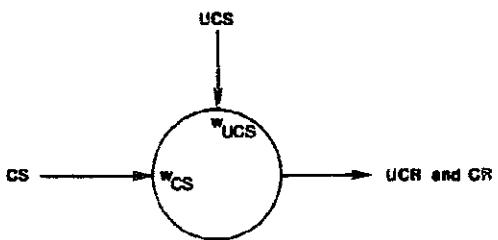


FIGURE 4.3. A Hebbian element as an analog of classical conditioning. The weight w_{UCS} associated with the UCS pathway is sufficiently large so that UCS occurrence causes the element to respond with the UCR. The weight w_{CS} of the CS pathway is initially too small for the CS alone to elicit a response, but increases with repeated simultaneous pairing of the CS and UCS until the CS alone can elicit a response - the CR.

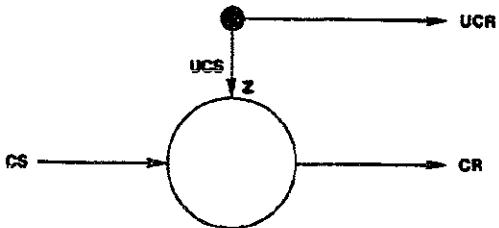


FIGURE 4.4. Some adaptive element analogs of classical conditioning require a specialized UCS pathway that causes modifications in the CS pathway but does not have an excitatory effect on the element. This implies that the UCR and CR pathways are separate so that stimulus substitution does not occur at the element. Additional assumptions must be made to account for the similarity of the UCR and CR.

then after sufficient simultaneous pairing of the UCS and CS, w_{CS} will increase to a value at which the CS will elicit a response, the CR, in the absence of the UCS.

One reason the Hebbian rule has remained influential among theorists is that it provides a very simple hypothesis to account for a stimulus substitution view of classical conditioning. It is a common, though not universally accepted, theoretical position that in classical conditioning, the CS comes to elicit a CR by effectively substituting for the UCS. This explains the similarity between the CR and the UCR since it implies that the two responses occur via the same response pathway's being activated by two different stimulus pathways. This view, known as stimulus substitution theory, has proved to be a reasonable generalization from the data (see discussion and review in Mackintosh, 1974, pp. 100-109). In the Hebbian model of classical conditioning (Figure 4.3), the CR and UCR share the same pathway so that one would expect them to be similar.

Other adaptive element analogs of classical conditioning do not provide so natural an account of the similarity between CR and UCR because they require the UCS to be a specialized input to the adaptive element that does not excite it (Figure 4.4). In these cases, separate

pathways are required for the CR and UCR. To account for the similarity of the CR and UCR it is necessary to postulate that the CR and UCR pathways converge in some manner "downstream" from the adaptive element. The perceptron of Rosenblatt (1962) and the informon of Uttley (1979) require this organization to form analogs of classical conditioning.

Aside from providing a simple explanation for the similarity of the CR and UCR, that the UCS is an unspecialized input in the case of the Hebbian element also means that the activity of any input pathway can cause changes in other pathways. In particular, pathways whose efficacies have become strengthened through previous training can further affect other pathways. A model with this property can produce behavior suggestive of higher order learning in animals: A previously conditioned CS can act as a UCS for a second CS. This property has also contributed to the interest in the Hebbian rule among theorists. It is not necessary to fix from the start the source of reinforcement. Any correlations among the input signals to an element will tend to be reflected in the connection weight values. The requirement for reinforcement to be provided only from a fixed source, on the other hand, raises the problem of somehow providing appropriate reinforcing signals at the appropriate times. The

significance of this problem may be reflected in the lack of success in constructing powerful adaptive networks of perceptron elements (see Minsky and Papert, 1969 and Minsky and Selfridge, 1961).

We present a new adaptive element analog of classical conditioning that uses the stimulus substitution organization shown in Figure 4.3. We briefly introduce the model here and discuss it in detail below. In addition to the stimulus signals x_i , $i = 1, \dots, n$, and the output signal y , our model requires the use of several other variables. First, for each stimulus signal x_i , $i = 1, \dots, n$, we require a separate stimulus trace which we denote by \bar{x}_i . By this we mean that the occurrence of CS_i at time t , indicated by $x_i(t) = 1$, initiates a prolonged trace given by nonzero values of separate variable \bar{x}_i for some period of time after t . This is accomplished by letting $\bar{x}_i(t)$ be a weighted average of the values of x_i for some time period preceding t . Similarly, we require a trace of the output y . Let $\bar{y}(t)$ denote a weighted average of the values of the variable y over some time interval preceding t . In the computer simulations which produced the data shown below, we generated these traces using the first-order linear difference equations

$$\bar{x}_i(t+1) = \alpha \bar{x}_i(t) + x_i(t) \quad (4.3)$$

$$\bar{y}(t+1) = \beta \bar{y}(t) + (1 - \beta)y(t) \quad (4.4)$$

where α and β are positive constants with $0 \leq \alpha, \beta > 1$. Appendix B gives the values actually used in the simulations.

The behavior of the adaptive element is therefore described by the values over time of the two variables y and \bar{y} , and the values of the three variables x_i , \bar{x}_i , and w_i for each input pathway $i = 1, \dots, n$. In terms of these variables, the model takes the form of a set of difference equations for successively generating the values of the associative strengths: for each i , $i = 1, \dots, n$,

$$w_i(t+1) = w_i(t) + c[y(t) - \bar{y}(t)]\bar{x}_i(t) \quad (4.5)$$

where c is a positive constant determining the rate of learning.

We can describe the process given by Equation 4.5 as follows: Activity on any input pathway i , $i = 1, \dots, n$, possibly causes an immediate change in the element output y but also causes the connection from that pathway to become "tagged" by the stimulus trace \bar{x}_i as being eligible for modification for a certain period of time (the duration of the trace \bar{x}_i). A connection is modified only if it is eligible and the current value of y differs from the value of the trace \bar{y} of y .

The effectiveness of the reinforcement for the conditioning process depends on the difference $y(t) - \bar{y}(t)$ which determines how the eligible connections actually change. The simplest case, and the one used in our simulations, results from letting $\beta = 0$ in Equation 4.4 so that $\bar{y}(t) = y(t - 1)$. Then $y(t) - \bar{y}(t) = y(t) - y(t - 1)$ which is a discrete form of the rate-of-change of the variable y .

Our use of stimulus traces to create periods of "eligibility" was borrowed from the neural hypothesis by Klopff (1972, 1981) that the temporal characteristics of conditioning, both classical and instrumental, can be produced if one set of conditions makes synapses eligible for modification of their transmission efficacies, but actual modifications occur due to other influences during periods of eligibility. This differs from related theories in that eligibility is seen as being indicated in some way completely separate from electrical activity. That is, instead of being marked as eligible for modification by a transient increase in efficacy, or by prolonged presynaptic activation, a pathway would be marked by some mechanism which does not participate directly in the electrical signaling of the cell, such as a transient increase in the concentration of a particular chemical.

The weight change rule given by Equation 4.5 can be roughly understood by analogy with the Hebbian rule. While the Hebbian rule detects correlations between input and output signals, this rule detects correlations between traces of input stimuli and changes in output. These differences have subtle and sometimes surprising consequences which will be discussed in the next three sections.

4.3 Temporal Relationships

The use of the stimulus traces \bar{x}_j and the output trace \bar{y} in our model permits it to reproduce some of the intratrial temporal relationships between stimuli and responses observed in classical conditioning experiments. Here we discuss interstimulus interval dependency and CR latency and review how earlier adaptive element models account for these aspects of classical conditioning. We then present simulation experiments which show that our model produces behavior in good agreement with experimental data.

We have said a pairing between the CS and the UCS is necessary for a classical conditioning association to form. In fact, many aspects of the temporal relationship between CS and UCS will affect the strength and rapidity of

conditioning. Both the words "pairing" and "associative learning" commonly used in reference to classical conditioning seem to imply a symmetrical relationship between the CS and the UCS, and many theorists have created models in which associations are formed when CS and UCS (or their theoretical analogs) occur simultaneously. Experimentally, however, simultaneous presentation of CS and UCS typically results in very poor conditioning, if any (e.g., Smith, Coleman and Gormezano, 1969).

An effective pairing of CS and UCS in classical conditioning is not a symmetric--the CS must occur first. The crucial variable with respect to the CS - UCS temporal relationship is the time interval between the onset of the CS and the onset of the UCS (the interstimulus interval, or ISI). Associative strength between the CS and the CR is usually found to be an inverted-U shaped function of this interval, being zero at simultaneous presentation, maximal at intermediate values (that depend strongly on the particular response system), and then falling toward zero at longer ISIs. Conditioning for negative ISIs, or backward conditioning, is generally considered not to occur (see Mackintosh, 1974, pp. 58-60). Figure 4.5 shows an example of this relationship.

A second important aspect of the intratrial temporal

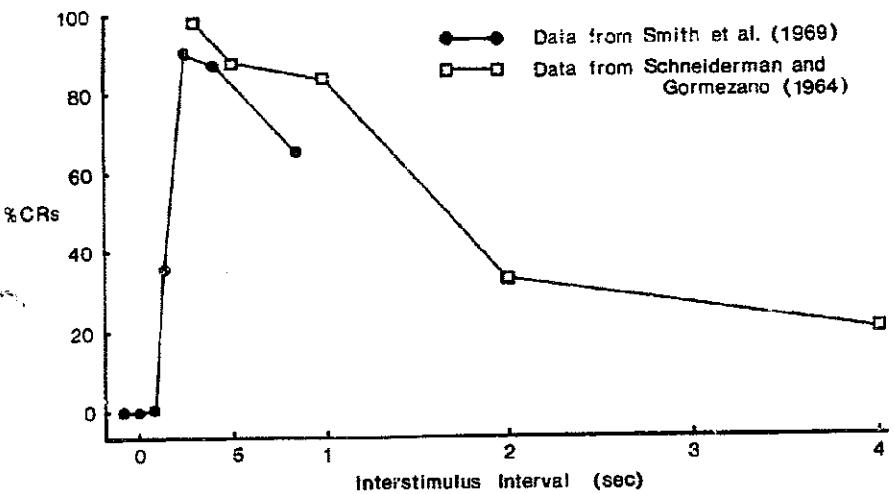
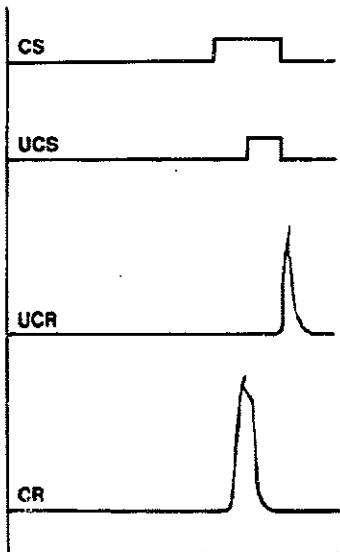


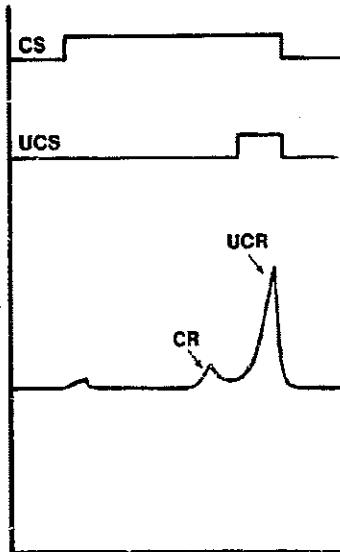
FIGURE 4.5. Asymptotic associative strength versus interstimulus interval in rabbit nictitating membrane response delay conditioning. Optimal ISI times vary widely from small fractions of a second for some response systems to up to a minute and perhaps longer for others.

relationships in classical conditioning is the time of occurrence of the CR relative to the CS and UCS. The time difference between CS onset and CR onset is called the CR latency. For a particular response there is usually a finite minimum value for the CR latency due to intrinsic delays of various kinds. For the nictitating membrane response, for example, the minimum CR latency is on the order of 70 - 80 msec. When the ISI is shorter than the minimum CR latency, then the CR necessarily begins after UCS onset. In the more usual case in which the ISI is longer than the minimum CR latency, the CR begins before the UCS (Mackintosh, 1974, p. 61). Two examples are shown in Figure 4.6.

In Figure 4.6a the CR begins nearly immediately after the CS, just as the UCR begins nearly immediately after the UCS. However, in many experiments, post-training behavior much like that shown in Figure 4.6b is observed, in which the CR begins much later than a minimum CR latency after CR onset. This appears to be the result of the animal discrimination between earlier and later parts of the CS and treating them as different CSs. (The CR initially begins soon after the overt CS onset and then gradually shifts later with continued training. This shifting is made more rapid by increasing the discriminability of earlier and later parts of the CS.) In these cases also the CR is



a) Leg flexion



b) Eyelid response

FIGURE 4.6. Tracings of CRs and UCRs in studies of leg flexion and eyelid conditioning. In each case CR onset occurs before UCS onset.

a) Leg flexion CR and UCR in dogs (after Kellogg, 1938).
b) Eyelid CR and UCR in a human subject (after Hilgard, 1936).

experimentally found to precede the UCS. Summarizing, we can state: Except in the case of an ISI less than the minimum CR latency, a classically conditioned CR will begin before its UCS (Mackintosh, 1974, p. 61).

It is on the basis of these temporal relationships that we say that the CS is a predictor of the UCS and the CR is a prediction of the UCS. Many learning theorists (e.g., Dickinson and Mackintosh, 1978, and Kamin, 1969) have emphasized the importance of the CS being an informative predictor of the UCS rather than just occurring appropriately paired with the UCS. To this we add that in order for the predictive information made available by the UCS to be useful, it must be available before the event predicted. This suggests that the fact that the CR occurs before the UCS in classical conditioning may be an important aspect of the classical conditioning behavior.

However, not one of the adaptive element models currently in the literature is capable of producing behavior whose temporal structure is in agreement with that observed in animal learning as described above. It is usual practice to add additional mechanisms, such as a delay in the CS pathway, in order to account for some of the temporal relationships between stimuli and responses. In most cases, however, the resulting adaptive elements display only

superficial aspects of this temporal structure.

4.3.1 Delays

As a first step it is important to understand what can, and what cannot, be achieved by the addition of delays in input and/or output pathways of elements requiring simultaneous pairing for changing weights. Consider the two different ways of using adaptive elements to model classical conditioning that we have described (Figures 4.3 and 4.4). These models differ in that the latter have a specialized UCS pathway and a UCR pathway that is different from the CR pathway.

First consider the consequences of adding a delay in the CS pathway in either type of model (Figure 4.7a). When the delayed CS temporally overlaps the UCS, the associative strength of the CS increases. This means that maximal learning occurs when the UCS follows the CS by the time of the delay, thus exhibiting a rough form of the experimentally observed ISI dependency. Suppose now that conditioning continues until the CS elicits the CR. Since the CS is delayed, the CR is also delayed, so that it cannot begin earlier than the UCS, i.e., the CR latency is always greater than or equal to the ISI. The delay in the CS pathway necessarily also delays the CR thus preventing it

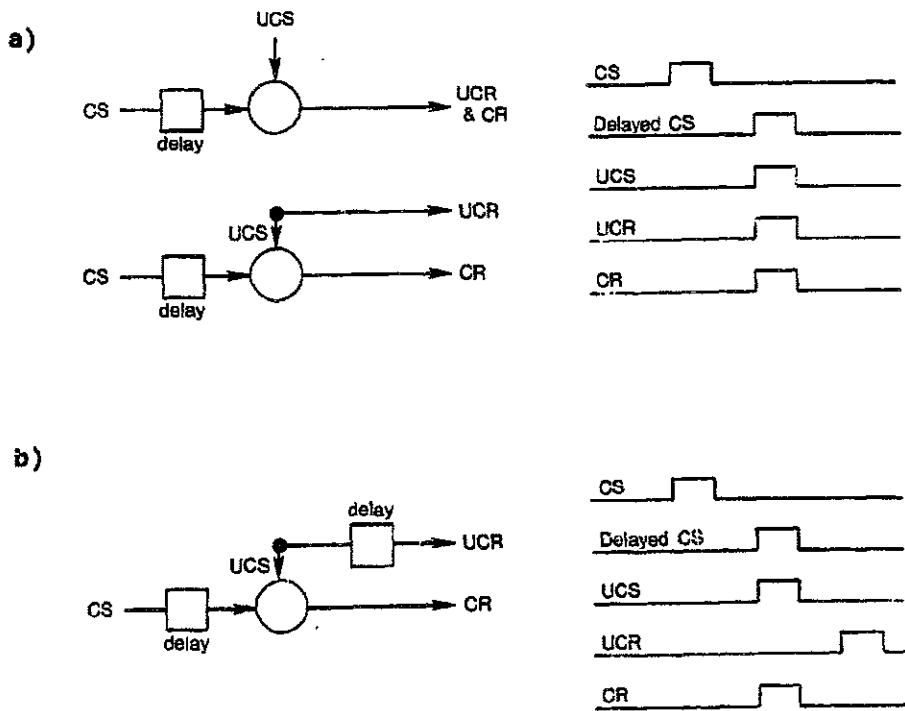


FIGURE 4.7. The use of delays in attempts to approximate the temporal relationships observed in classical conditioning.
 a) A delay in the CS pathway of both types of classical conditioning models necessarily also delays the CR.
 b) Delays in the CS and UCR pathways permit the CR to precede the UCR but not the UCS.

from being a useful prediction.

For the case in which there are separate pathways for the CR and the UCS (Figure 4.4), one can consider adding delays to both the CS and the UCR pathways as is done, for example, by Uttley (1975). In this case, the CR cannot occur earlier than the UCS for the same reason discussed above, but it can occur earlier than the UCR due to the delay in the UCR pathway (Figure 4.7b). However, in classical conditioning it is the UCS that is anticipated by the CR. That is, an animal can predict stimuli by becoming sensitive to external signals which regularly precede those stimuli. Merely producing a response earlier than it previously appeared, but not before the previously eliciting stimulus, simply results in increased speed of response. This is indeed a useful strategy, but it can be accomplished more simply by reducing the delay in the UCR pathway. In classical conditioning, on the other hand, a response can occur earlier than the occurrence of the stimulus which previously elicited it. This, of course, requires the availability of predictive information in the environment (a CS).

For elements requiring simultaneous pairing of stimuli for forming associations, no combination of simple delays in the CS, UCS, and UCR pathways can produce this kind of

anticipatory response. The delays essentially just slow the system down. In addition, delays on the order of seconds or even longer required for this approach are very hard to justify neurophysiologically.

4.3.2 Stimulus Traces

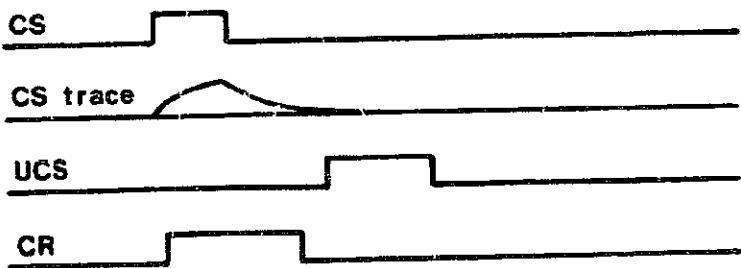
The notion that a stimulus sets up an internal neural trace which persists after the stimulus ends has a long history in theories of learning, notably in Hull's (1943), and has been used in neural network theories as, for example, by Grossberg (1974). Although a simple delay is one form of stimulus trace, the kind of trace to be considered now is one which, unlike a delayed signal, persists in some form throughout the temporal interval. In particular, such a trace is present in the interval's early as well as late portions. There are two general classes of possibilities for stimulus trace mechanisms: 1) traces are maintained by the firing levels of some neurons, possibly by means of reverberatory circuits, and 2) they are maintained by something other than neuronal electrical activity, perhaps by chemical concentrations. From our theoretical point of view, the most important difference between these two possibilities is that the former employs the same means for storing traces as is used for signaling stimuli and producing responses. In the latter case, these two

functions are performed by separate mechanisms. The first type of trace, which we call a stimulating trace, is more frequently hypothesized, and we discuss this possibility first.

Suppose the CS gives rise to a stimulating trace which persists long enough to span the interval between CS and UCS presentations (Figure 4.8a). If this trace serves as the CS input to an adaptive element requiring simultaneous pairing, and the UCS does not produce such a trace, one can obtain an ISI dependency curve whose shape resembles that of the stimulus trace function [footnote]. If the UCS leaves a similar stimulus trace that acts as input to the adaptive element, then the ISI dependency curve shows substantial learning for negative CS-UCS intervals, i.e., for cases in which the UCS precedes the CS (Figure 4.8b). Uttley (1975)

Hull (1943) apparently believed that an experimental ISI curve could be accounted for by assuming a neural trace of the same shape. As Hilgard and Bower (1975) point out, however, level of conditioning is such a complex function of the ISI along with many other factors that this form of explanation is untenable. It should be noted, though, that since we are discussing adaptive elements out of which adaptive networks can presumably be constructed, this objection holds less force. The externally observed behavior of a network would be a product of the interaction of a variable mixture of local traces.

a)



b)

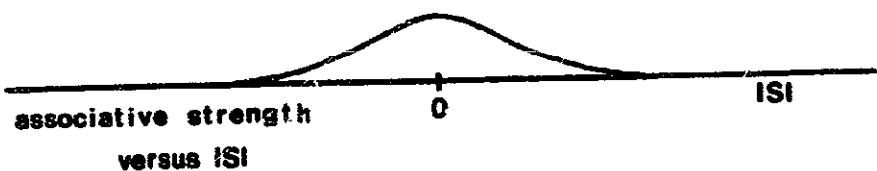
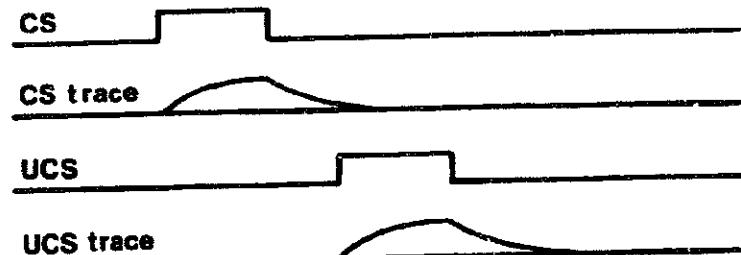


FIGURE 4.8. Stimulating stimulus traces.
 a) If the CS initiates a prolonged stimulating trace and the UCS does not, then the CR can anticipate the UCS, but the CR will tend to be prolonged also unless some additional mechanism is postulated.
 b) If both the CS and the UCS initiate traces which stimulate one of the adaptive elements described in the text, then there will always be backward conditioning. Shown here is an ISI dependency curve for the case in which the CS and UCS produce identically decaying exponential traces.

suggests the use of a long CS trace and a short UCS trace in order to minimize (but not eliminate) backward conditioning.

A stimulus trace consisting of a prolonged CS signal does permit the CR to anticipate the UCS since the signal trace, unlike a delayed signal, is present at the beginning as well as the end of the ISI. For example, if we assume that an element produces a response whenever the weighted sum of its input signals exceeds a threshold, then after sufficient training, the CS will elicit a CR whenever the CS trace, multiplied by the connection weight of the CS pathway, exceeds the threshold (Figure 4.8a). As training continues one would expect the duration of the CR to lengthen as longer intervals of the stimulus trace exceed threshold. Although various characteristics of the CR change as training continues, there are no data indicating a tendency for the CR to persist throughout the ISI: The CR generally resembles the UCR. Some additional mechanism would have to be postulated to prevent the prolonged stimulating trace from being manifested in overt behavior as a prolonged response.

4.3.3 Non-Stimulating Traces

We now consider what one would expect if the stimulus trace were provided by a signal different from the

stimulating signal. Several proposed mechanisms fall into this category. It has been suggested, for example, that a stimulus might leave a temporarily persistent trace in the form of an altered threshold of the postsynaptic element (Milner, 1957; Rosenblatt, 1962, p. 55), or that a transient increase in synaptic efficacy follows presynaptic activity and is made more permanent by subsequent firing of the postsynaptic cell (Rosenblatt, 1962, p. 57).

The use of a stimulus trace variable entirely separate from the major signaling variable has been proposed by Klopff (1972, 1981). He suggests that when activity at a synapse satisfies certain criteria, then that synapse becomes eligible for modification and remains eligible for a period of several seconds. The extent to which an eligible synapse is modified depends on the reinforcement level during the period of eligibility. Each synapse is therefore viewed as possessing its own local trace mechanism which mediates synaptic modification but does not directly alter any other aspect of the unit's behavior. Such a trace can persist, as Klopff suggests, for the relatively very long times suggested by classical (and instrumental) conditioning data without interfering with ongoing signal transmission. Further, the large variation in ISI dependency for different response systems might be accounted for by variations in eligibility trace durations. This is the kind of stimulus trace

provided by the term \bar{x}_i in our model.

Our model implies that a synapse becomes eligible for modification whenever a presynaptic signal occurs there, and that eligibility forms a curve like that of the trace in Figure 4.8a (see also Figure 4.10). Since learning occurs due to an interaction between the UCS signal and a non-stimulating eligibility trace initiated by the CS, the critical temporal aspects of classical conditioning can be produced. In particular, the CR will begin immediately after the CS and, unlike the case of a stimulating trace, the CR will not extend in duration as conditioning proceeds (see below). This is possible because the trace is different from the stimulating signal.

Although both stimulating and non-stimulating traces might be postulated to account for the important temporal aspects of classical conditioning, a non-stimulating trace has the advantage of permitting a clear distinction to be maintained between actual stimuli and traces of stimuli. There are two countervailing requirements that need to be met. First, fast electrical signals are necessary to indicate as precisely as possible the time of occurrence of specific events. It is to an organism's advantage to perceive events as occurring as closely as possible to their actual time of occurrence, and particularly as early as

possible. Second, it is necessary to retain the knowledge of these occurrences so that they can be associated with later events. In a two variable system, these two requirements are both satisfied whereas in a single variable system, such as one using reverberatory activity, one of these requirements can only be satisfied at the expense of the other. If the association of events depends on their precise temporal relationship, as indeed it appears to, then we can expect there to be a high priority on precise temporal localization of events. Thus, it seems most reasonable not to confuse the need for a short distinct signal with the need for a prolonged trace by using a single trace for both purposes.

A common argument for a reverberatory activity theory is based on certain studies of attention and distraction and their effect on learning. These studies indicate that reverberatory activity is probably important in the central nervous system. However, this we do not mean to debate. Reverberatory activity can be expected to play an important role--for example, it can determine what information is picked up or relayed to higher centers. We believe it is unwarranted, however, to proceed from this to the conclusion that reverberatory activity is the primary mechanism for spanning the time between the sequential events upon which learning is contingent.

4.3.4 Model Behavior in Classical Conditioning with a Single CS

One uninteresting steady state of our model occurs when all the connection weights are equal to zero. In this case y remains at zero so that no modifications to the weights can occur. A simple way to exclude this steady state is to set at least one weight to a fixed nonzero value. In an analog of classical conditioning, this fixed input pathway carries the UCS, and the resultant effect on the element is the UCR (see Figure 4.9).

It is useful to consider the simplest special case of a single rectangular CS signal which ends when the UCS starts. The discussion is also simplified if we assume that the UCS is sufficiently long so that all synapses have lost their eligibility by the time of its offset. Figure 4.10 shows this CS, the eligibility it generates, as well as a UCS and the reinforcement signal generated. We have assumed that w , the associative strength of the CS, is initially equal to zero and that the term \bar{y} takes the simplest form $\bar{y}(t) = y(t-1)$ resulting from letting $\beta = 0$ in Equation 4.4. This makes $y - \bar{y}$ a rough form of the derivative of y . The rectangular CS signal causes an increase in the eligibility of the CS pathway which persists for some time after the CS offset. The rectangular UCS signal, active through a fixed excitatory input of strength λ , causes a positive change in

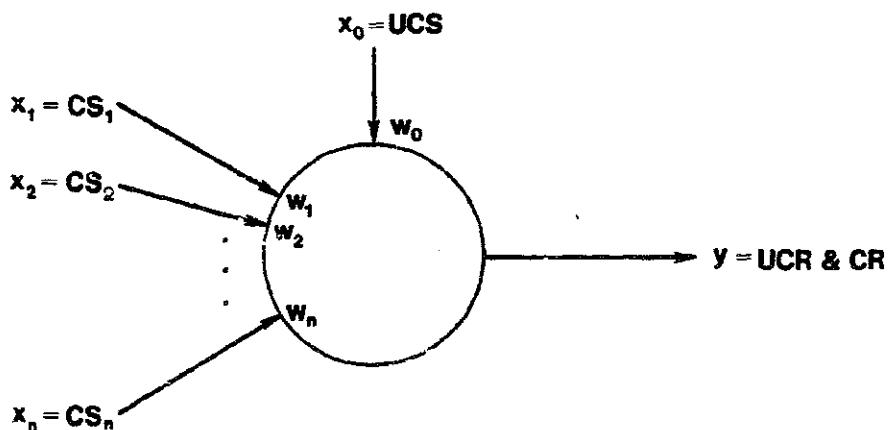


FIGURE 4.9. Our adaptive element as an analog of classical conditioning. There are n modifiable CS input pathways and a pathway with fixed weight w_0 which carries the UCS. The element output y represents both the UCR and the CR.

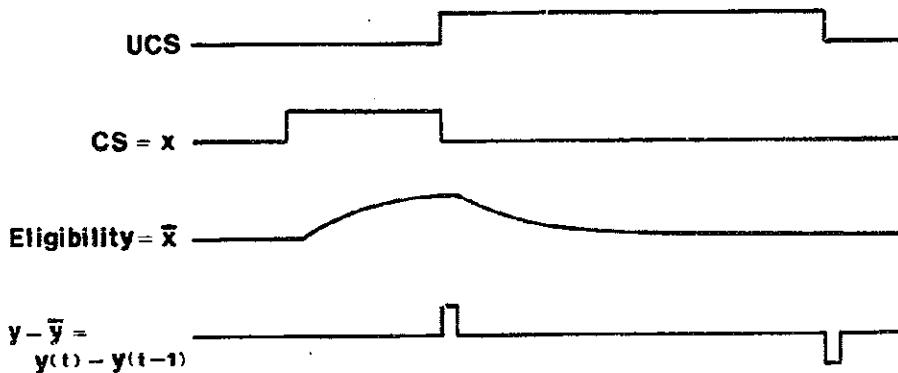


FIGURE 4.10. Time courses of element variables for a trial in which a neutral (associative strength $w = 0$) CS is followed by a UCS. For ease of explanation CS offset and UCS onset coincide, and the UCS is of sufficient duration so that \bar{x} is zero at UCS offset. The trace \bar{x} of the CS signal x indicates the eligibility for modification of the CS pathway. This trace increases during CS presentation and persists after CS offset. Element output y shows no change during CS presentation since $w = 0$, but since the UCS stimulates the element via a fixed positive weight, the shape of the time course of y follows that of the UCS signal. This causes $y - \bar{y}$ to indicate UCS onset with a positive pulse and UCS offset with a negative pulse. The CS associative strength w changes according to the product of \bar{x} and $y - \bar{y}$. Consequently, w increases at UCS onset and decreases by a lesser amount (here, by zero) at UCS offset, thus experiencing a net increase.

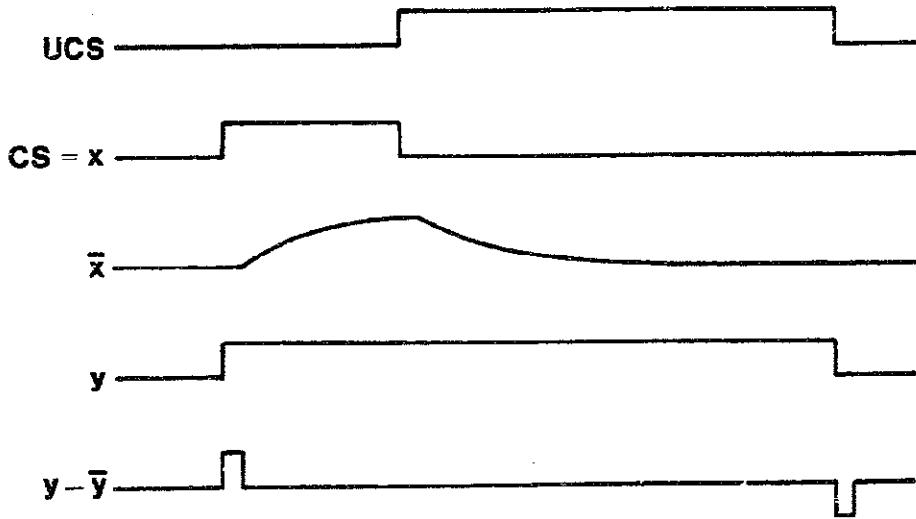


FIGURE 4.11. Time courses of element variables after the asymptotic CS associative strength has been reached due to a series of trials. Element output y changes at CS onset since w is now positive. UCS onset causes no additional increase in y over that level produced by the CS. The CS pathway eligibility \bar{x} is zero for the positive pulse of $y - \bar{y}$ and, assuming a sufficiently long UCS, also zero for the negative pulse. Under these circumstances, w does not change.

y at its onset and an equal but negative change at its offset. The weight, or associative strength, of the CS experiences a net increase: At the UCS onset it increases by a certain amount and decreases by a lesser amount at the UCS offset (in this case the decrease is zero since the eligibility has decayed to zero by the time of UCS offset).

After one trial, w is positive so that on the next trial the occurrence of the CS increases the output level y . Consequently, CS onset causes a transient increase in $y - \bar{y}$ that has no effect on the CS pathway since CS pathway eligibility is zero at CS onset. However, the level of y is raised by the CS so that UCS occurrence causes less of an increase in y than it did on the preceding trial. This means that the value of $y - \bar{y}$ at the time of UCS onset causes a further increase in w , but one of smaller magnitude than in previous trials. With additional trials, this process continues until the value of $y - \bar{y}$ at the UCS onset is equal to zero, that is, until the CS produces activity equal to that produced by the UCS (Figure 4.11). Growth in associative strength therefore is negatively accelerated and stops when y remains constant during CS pathway eligibility. Figure 4.14, trials 0-10, shows the form of the acquisition curve produced by computer simulation.

The equilibrium reached after a number of trials and

shown in Figure 4.11 has the following important properties. First, the CS has an excitatory effect on the adaptive element when the effect of the UCS is also excitatory. This permits a stimulus substitution model of classical conditioning in which the CR and UCR share the same pathway (Figure 4.3). Second, the CR produces an output level y of magnitude equal to that produced by the UCS. Third, the CR is produced earlier than the UCS. The element increases its output level in anticipation of UCS occurrence.

Similar behavior is produced when UCS onset precedes or follows CS offset by some time interval or when the eligibility trace outlasts the UCS. In these cases, however, the CR will differ in magnitude from the UCR in a manner depending on the precise temporal arrangement of the CS and UCS. In addition, the equilibria in these cases are dynamic rather than static. The CS associative strength continues to change during each trial, but eventually there is zero net change per trial. The behavior approaches a stable limit cycle. Appendix A contains a related formal analysis.

Figure 4.12 shows the resultant asymptotic connection weight for a series of simulation experiments in which the time interval between CS onset and UCS onset is varied. The connection weight becomes the strongest when the CS ends

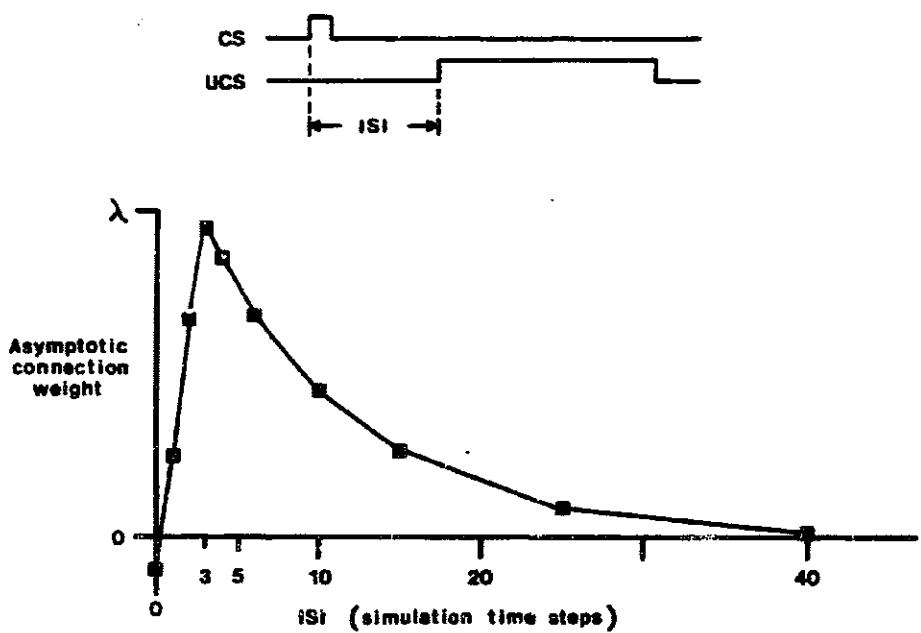


FIGURE 4.12. Asymptotic connection weight versus interstimulus interval in a simulated classical conditioning paradigm. The interstimulus interval (ISI) was varied between 0 and 40 time steps, CS length was 3 time steps, and UCS length was 30 time steps.

just as the UCS begins ($ISI = 3$ time steps). At ISIs less than 3 time steps there is less time for the eligibility of the CS pathway to increase before the arrival of the UCS. At ISIs greater than 3 intervals the eligibility decays toward zero since the CS is not present for some interval between CS offset and UCS onset. These results have the same overall form as those observed in animals.

However, in animal experiments optimal ISIs are not so strongly tied to overt CS duration, although longer optimal ISIs have been observed for long fixed delay CSs than for short trace CSs (Schneiderman, 1966). The behavior of our adaptive element can be reconciled with the experimental observations if it is assumed that "effective", or "internal" CS duration is not identical to overt, external CS duration. A long CS is ignored shortly after it begins, while even an instantaneous overt CS causes an internal representation of some significant duration. This internal duration, rather than overt CS duration, then, would determine optimal ISI.

Behavior similar to that discussed above is produced by our model if \bar{y} is a more prolonged trace than that used for the preceding discussion. Letting β be nonzero (but still less than one) in Equation 4.4 results in an exponentially decaying trace \bar{y} similar to the eligibility trace \bar{x} . In

this case, the term $y - \bar{y}$ used in our model is a measure of the deviation of the current output level from an average of past values. The low pass filtering characteristic of this measure prevents high-frequency fluctuations in y from significantly influencing the associative strengths. Equation 4.4 implies that for any β , $0 \leq \beta < 1$, if y remains constant over time, then $y - \bar{y}$ will approach zero, thus providing for deceleration of the learning process in a manner qualitatively similar to that produced when $\beta = 0$.

These illustrations of our model's behavior show that it is sensitive to the temporal relationships between stimuli within classical conditioning trials and is capable of producing CRs that occur before the UCS. It is evident from our discussion of how these properties follow from Equations 4.3, 4.4, and 4.5 that considerable behavioral subtlety can be generated by the interaction of eligibility traces and a measure of output change. In general, the quantitative aspects of our model's behavior depend on the timing, durations, and shapes of the CS and UCS signals, the forms of the eligibility traces \bar{x}_i and the output trace \bar{y} , and the character of the output mapping function f . This complex of dependencies provides considerable latitude for making quantitative predictions about particular response systems, and we restrict our attention in this report to the qualitative aspects of the model's behavior. Appendix A

contains a mathematical analysis of some of these dependencies for a simplified version of the model.

4.4 Context and Expectation

Another aspect of classical conditioning which should be included in even a very simple theory is the effect of the context of a CS. The associative strengths of the stimuli that act as context for a CS on a trial can nullify or even reverse the effect of the occurrence of the UCS on that trial. This can be seen in numerous experimental paradigms, of which the simplest is known as blocking.

In blocking, as in all stimulus context experiments, a compound stimulus consisting of at least two stimulus components (one of which is frequently thought of as a conglomerate background stimulus component) is used as a CS. In part I of a typical blocking experiment one stimulus component CS_1 , which might be a light, is paired with a UCS at an appropriate ISI until associative strength between CS_1 and the CR reaches its asymptotic value. In part II, the experimenter continues to pair CS_1 with the UCS, but also pairs CS_2 , say a bell, with identical temporal relationship as diagrammed in Figure 4.13. In effect, the compound stimulus $CS_1 + CS_2$ is being paired with the UCS.

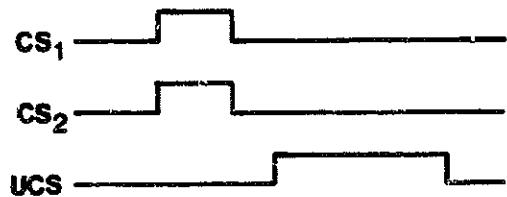


FIGURE 4.13. Temporal relationship between stimuli in the second part of a standard blocking experiment. Learning to each of the component stimuli CS₁ and CS₂ will depend on the associative strength of the other component stimulus.

The result of this procedure is that CS₂, which is appropriately paired with the UCS in part II, conditions very poorly, if at all, compared to a control group without part I conditioning to CS₁. This is not an isolated result. Effects of the associative strengths of context stimuli on conditioning occur in a great variety of experimental paradigms, in both classical and instrumental conditioning, of which blocking, overshadowing, and conditioned inhibition are only some of the more prominent examples (see Hilgard and Bower, 1975, pp. 571-573). Context stimuli can have such large effects on resultant associative strength that they cannot satisfactorily be ignored by a nontrivial theory of classical conditioning.

The simplest and most successful theory describing the effects of stimulus context is generally considered to be that of Rescorla and Wagner (1972). They state their theory in cognitive terms as follows:

...organisms only learn when events violate their expectations. Certain expectations are built up about the events following a stimulus complex; expectations initiated by the complex and its component stimuli are then only modified when consequent events disagree with the composite expectation. (p. 75)

Applying this analysis to the blocking experiment: Part I builds up an expectation that the UCS will follow CS₁. The events of part II do not violate this expectation, so there is no learning. Other stimulus context effects can be dealt

with in similar fashion. However, similar ideas have been advanced by others. What distinguishes Rescorla and Wagner's theory is that it is given a precise mathematical form:

$$\Delta V_A = \alpha_A \beta [\lambda - V_{AX}] \quad (4.6)$$

where ΔV_A is the change in associative strength to a CS A, λ is the asymptotic value of associative strength possible with the UCS, V_{AX} is the associative strength already present to the stimulus complex A + X, where X is a conglomerate background stimulus, and α_A and β are positive constants depending respectively on the CS being changed (A) and the particular UCS used. Implicit here is that Equation 4.6 is only applied to a CS A if it is present on the trial, and that the complex A + X is precisely all stimuli present on the trial. Using the simplest assumption that $V_{AX} = V_A + V_X$, taking $c = \alpha_A \beta$, and letting S be the set of (indexes of) all stimuli present on a trial, Equation 4.6 can be written as

$$\Delta V_{CS_i} = \begin{cases} c[\lambda - \sum_{j \in S} V_{CS_j}] & \text{for } i \in S \\ 0 & \text{for } i \notin S. \end{cases} \quad (4.7)$$

Part I of the blocking experiment results in V_{CS_1} reaching the value λ because CS_1 is the only stimulus present. In part II, $V_{CS_2} = 0$ initially, and since

$$\sum_{j \in S} V_{CS_j} = V_{CS_1} + V_{CS_2} = \lambda + 0 = \lambda,$$

no changes in associative strength take place. It should be clear how this equation implements Rescorla and Wagner's cognitive theory referred to above: The expectations that are built up are the associative strengths, and these are modified when events such as the UCS, represented by λ , differ from the composite expectation (the sum of the associative strengths of the stimuli present).

This theory can account for blocking and a wide range of the other stimulus context effects. The theory is not a completely satisfactory one, the two most prominent and best established shortcomings being: 1) There has been repeated failure to demonstrate the extinction of conditioned inhibitors predicted by the Rescorla-Wagner model (the return to zero of negative associative strengths when their stimuli occur without any correlation to the UCS; Zimmer-Hart and Rescorla, 1974). 2) The strict application of the Rescorla-Wagner equation requires the prediction of a strictly negatively accelerated acquisition curve. The consensus is that this curve is initially positively accelerating (Mackintosh, 1974, p. 11). The Rescorla-Wagner theory also does not correctly predict the microstructure of individual response sequences (Prokasy and Gormezano, 1979). Recent extensions to the Rescorla-Wagner model have been proposed to remedy some of these problems (Frey and Sears, 1978).

Our adaptive element uses a form of expectation closely related to that of the Rescorla-Wagner model. Whereas in that model the associative strengths are changed based on the difference between received and expected UCS levels, in our model weights are changed based on the difference between actual activity level y and expected activity level \bar{y} . In fact, our model results in all the stimulus context behavior of the Rescorla-Wagner model.

This can be seen most clearly by considering another special case. Assume there are many CS pathways, on which rectangular pulse CSs may or may not be present, and that all CSs present on a trial begin simultaneously (and, as before, end as the UCS begins). If the UCS signaled by x_0 begins at time T and has a duration longer than the eligibility traces, then the connection weight w_i , corresponding to CS_i , can only change at T . This is the only time at which y changes when an input pathway can be eligible. Then the total change in w_i on a particular trial is $\Delta w_i(T)$. From Equation 4.5 we have:

$$\Delta w_i(T) = c[y(T) - \bar{y}(T)]x_i(T)$$

Taking the simplest case $y(t) = y(t-1)$, and $\bar{x}_i(t) = x(t-1)$:

$$\Delta w_i(T) = c[y(T) - \sum_{j=0}^n w_j x_j(T-1)]x_i(T-1).$$

Letting $y(T) = w_0x_0(T) = \lambda$, and noting that $x_0(T - 1) = 0$, we obtain

$$\Delta w_i(T) = c[\lambda - \sum_{j=1}^n w_j x_j(T - 1)]x_i(T - 1).$$

And since $x_j(T - 1) = 1$ indicates CS presence, we can write

$$\Delta w_i(T) = \begin{cases} c[\lambda - \sum_{j \in S} w_j] & \text{for } i \in S \\ 0 & \text{for } i \notin S \end{cases}$$

where S is the set of stimuli present on the trial. Since $\Delta w_i(T)$ is the total change in connection weight on the trial, this result is identical to the Rescorla-Wagner equation (Equation 4.7).

Computer simulations illustrate this result in a variety of standard stimulus context experiments. The results of a computer simulation of our model in a blocking experiment is illustrated in trials 0-20 of Figure 4.14. For the first 10 trials of the simulation experiment CS_1 is presented alone and followed by the UCS as discussed earlier. The connection weight w_1 of CS_1 quickly rises to the UCS level $\lambda = .6$ (see Figure 4.14, trials 0-10; Figure 4.11 shows the steady state element behavior; additional details on the simulations are in Appendix B). The acquisition curve is purely negatively accelerated as in the Rescorla-Wagner theory.

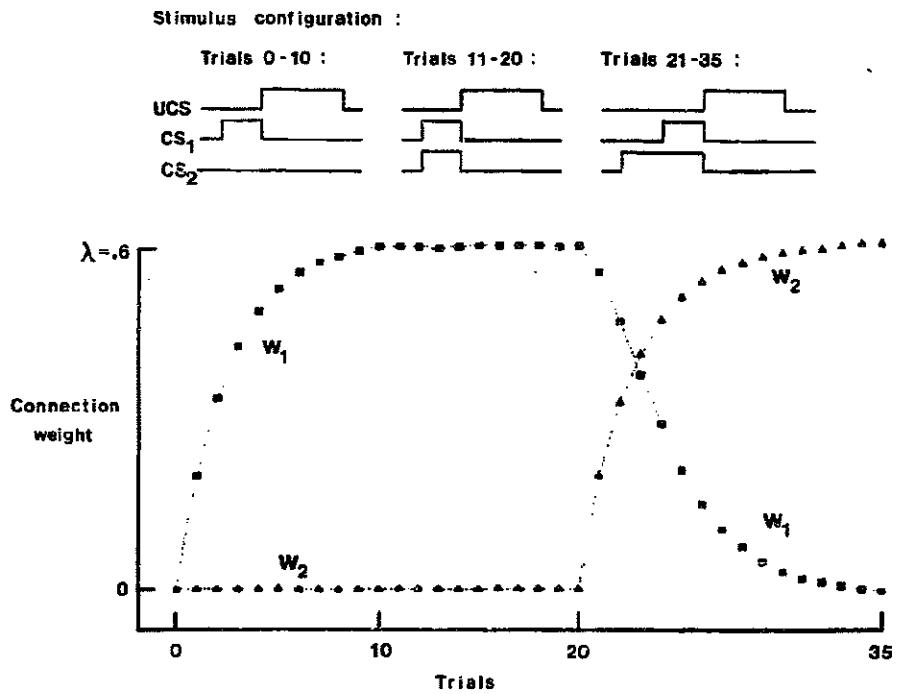


FIGURE 4.14. The connection weights at the end of each trial in a simulation experiment. The intratrial time courses of the variables involved are not shown.
 Trials 0-10: Presentation of CS₁ alone followed by the UCS results in w_1 increasing.
 Trials 11-20: CS₁ and CS₂ presented together followed by the UCS produces no change since CS₂ is redundant. This is the blocking paradigm.
 Trials 21-35: CS₂ begins earlier than CS₁. The element becomes sensitive to the earlier predictor and loses sensitivity to the later.

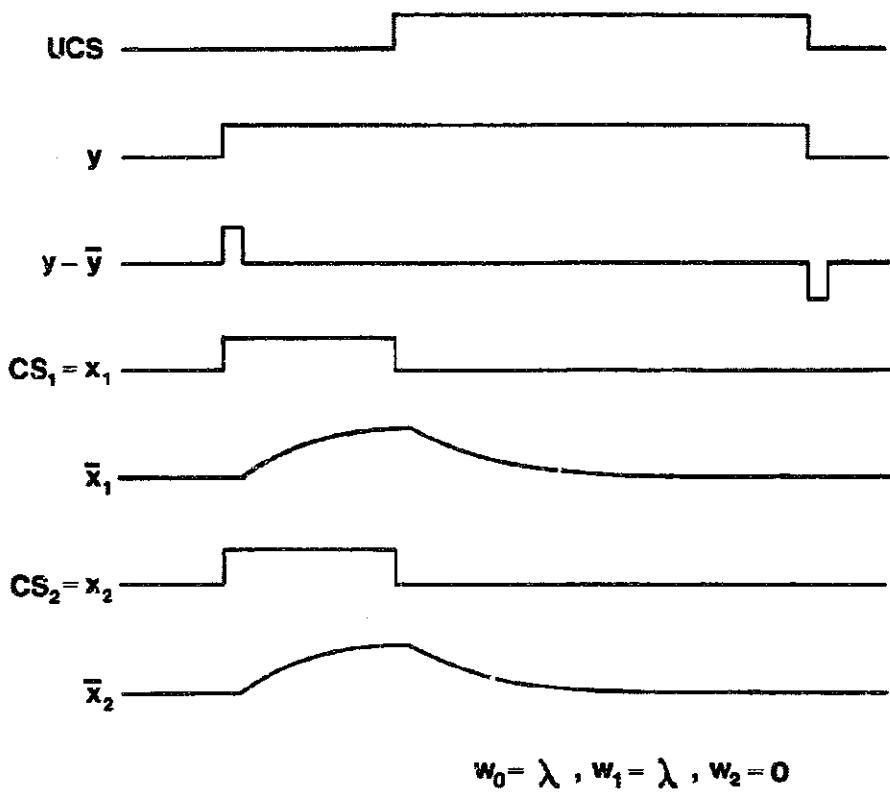


FIGURE 4.15. Intratrial time courses of element variables in part II of a blocking experiment (trials 11-20 shown in Figure 4.14). Since the weight associated with CS_1 has already reached its asymptotic value of λ , $y - \bar{y}$ is zero whenever CS_2 pathway eligibility \bar{x}_2 is nonzero. Consequently, no changes in weight values occur.

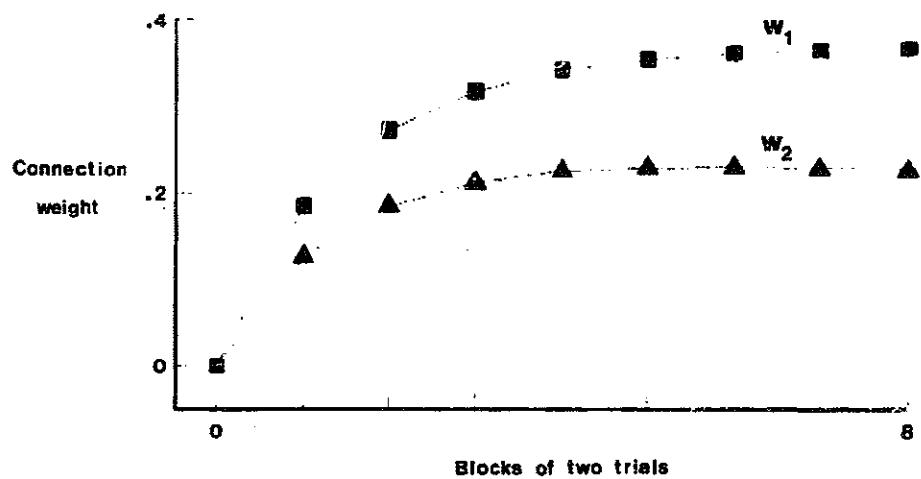


FIGURE 4.16. Simulation results of an experiment with two CSs each of which accounts for a particular portion of the UCS's reinforcement. Trials of CS_1 paired with a UCS of strength $\lambda = .4$ were alternated with trials in which the compound $CS_1 + CS_2$ was paired with a UCS of strength $\lambda = .6$.

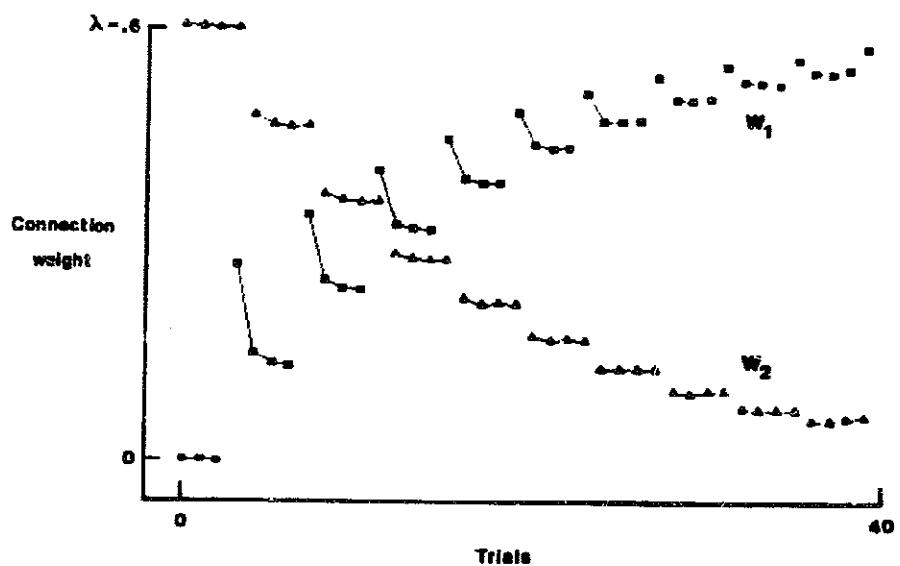


FIGURE 4.17. Simulation results of an experiment with two CSs differentially associated with the UCS. CS₁ precedes every UCS whereas CS₂ is absent every fourth UCS. Although initially CS₂ is dominant ($w_1 = 0$, $w_2 = \lambda$), eventually CS₁, the more reliably associated CS, dominates ($w_1 = \lambda$, $w_2 = 0$).

For trials 11-20, CS_1 is presented identically paired with CS_2 , and both are followed by the UCS. This is the blocking paradigm. Since it provides no new information about UCS arrival, CS_2 is redundant. During these trials w_1 and w_2 do not change. This result can be understood by examining the behavior of the relevant element variables during one of these trials (Figure 4.15). The decrease in y occurs too long after the occurrence of the CSs for them to be still eligible, and the increase in y occurs just as the CSs begin, and thus before they are eligible.

Elements that implement the Rescorla-Wagner equation find input signals whose presence is associated with the UCS and that are not redundant. Each such signal generates an expectation equal to the additional UCS magnitude indicated by its presence. If there are many signals, the sum of their expectations is of appropriate magnitude. For example, if the compound stimulus $CS_1 + CS_2$ is paired with a UCS of strength $\lambda = .6$, while CS_1 alone is concurrently paired with a UCS of strength $\lambda = .4$, then the two associative strengths (connection weights) w_1 and w_2 will stabilize at .4 and .2 respectively (assuming $x_i = 1$ indicates CS_i present; in general w_1x_1 and w_2x_2 will stabilize at .4 and .2). A simulation experiment confirmed this conclusion for the adaptive element we have introduced (Figure 4.16).

Elements that implement the Rescorla-Wagner equation also have a tendency to find the input pathways whose activity is most reliably associated with the UCS and to ignore all others. For example, let CS_1 be paired with 100% of the UCSs while CS_2 is paired with only 75% of the UCSs. Even if CS_2 is initially dominant in terms of associative strength ($w_1 = 0$, $w_2 = \lambda$), eventually CS becomes completely dominant ($w_1 = \lambda$, $w_2 = 0$). This result contrasts strongly with the blocking experiment in which equally reliable CSs do not change their dominance relation (Figure 4.14, trials 11-20). A simulation of our element in this situation produced results shown in Figure 4.17.

These simulations confirm that when viewed at the trial level and given the assumptions made above, our model behaves as the Rescorla-Wagner model, and, in particular, produces the stimulus context effects of that model. When viewed at the level of trials our model also shares the shortcomings of the Rescorla-Wagner model regarding extinction of conditioned inhibitors and the shape of the acquisition curve. Extensions of the Rescorla-Wagner model proposed to eliminate these shortcomings (Frey and Sears, 1978) are also applicable to our model. However, even with these extensions, the Rescorla-Wagner model applies only at the level of trials. It cannot supply predictions about the effects on conditioning of the intratrial temporal

relationships between stimuli. As we have seen, our model does apply to this intratrial structure for the case of a single CS, having behavior consistent with data on CR latency and ISI dependence. In addition, our model provides an extension of the Rescorla-Wagner use of expectation to a form having meaning within trials. This leads to several novel and interesting forms of model behavior.

The adaptive element we have presented finds the earliest predictors and ignores redundant later predictors. A CS that arrives simultaneously with, or after, a UCS is useless as a predictor. By the same reasoning, predictors that occur earlier than others are in some sense more predictive and potentially more useful. A later predictor can be redundant to an earlier one in the same sense that an unreliable predictor can be redundant to an identically timed but reliable predictor. For example, let CS_1 and CS_2 both always be followed by reinforcement, but let CS_2 start earlier than CS_1 . Then even if initially CS_1 is dominant ($w_1 = \lambda$, $w_2 = 0$), eventually CS_2 , the earlier predictor, will completely dominate CS_1 as a predictor of the UCS (eventually $w_1 = 0$, $w_2 = \lambda$). The result of a simulation of this experiment is shown as trials 21-35 of Figure 4.14 (recall that at trial 20, $w_1 = \lambda$ and $w_2 = 0$). Although both stimuli are being presented in trials 11-20 and in trials 21-35, in the former trials, CS_2 is blocked by

CS₁, while in the latter, the associative strength of CS₂ increases quickly and CS₂ comes to completely dominate CS₁. In the earlier trials, CS₂ is redundant to CS₁, which had already been conditioned, but in these later trials CS₂ provides important new information: It is the earliest indicator that the UCS will occur. This advantage, combined with the fact that CS₁ is totally redundant to CS₂, produces complete conditioning to CS₂ and the elimination of conditioning to CS₁.

This steady state is approached quickly and in an orderly manner, but the reasons for this behavior are somewhat difficult to explain. Very briefly, on each trial the associative strength w_2 of CS₂ increases and then decreases by a lesser amount for a net gain, while the associative strength w_1 of CS₁ only decreases: w_2 increases because CS₂ predicts the onset of CS₁'s excitation, and both w_1 and w_2 decrease at the offset of CS₁ and CS₂ because these two stimuli together produce too much expectation.

Although this property of the adaptive element to become sensitive only to earliest predictors of a UCS when the later ones provide no new information is reminiscent of some learning theory results (notably the work of Egger and Miller, 1962, on conditioned reinforcement), our primary interest in it stems from adaptive systems considerations.

We feel that a simple mechanism which finds the earliest, most reliable, and nonredundant predictors of important events is potentially very useful for constructing powerful adaptive systems.

4.4.1 Higher Order Conditioning

Although much of the discussion has been in terms of fixed pathways (corresponding to UCSs) causing changes in modifiable pathways, signals on these modifiable pathways, since they also can affect y , can also cause such changes. The simplest example of this corresponds to what is known as higher order conditioning in animal learning theory. A signal on a modifiable pathway (CS_1) is paired with a fixed input (UCS) until the connection weight w_1 reaches its asymptotic value. Then a signal on a second modifiable pathway (CS_2) is paired with a signal on the first modifiable pathway (CS_1). In this second pairing CS_1 acts as a reinforcing UCS for CS_2 . With repeated pairings the second connection weight w_2 grows to the level of w_1 , but w_1 , since its use is not followed by a UCS, gradually falls to zero. The result is that w_2 rises to the level of w_1 , and then follows w_1 to zero. The results of a simulation of this experiment are shown in Figure 4.18.

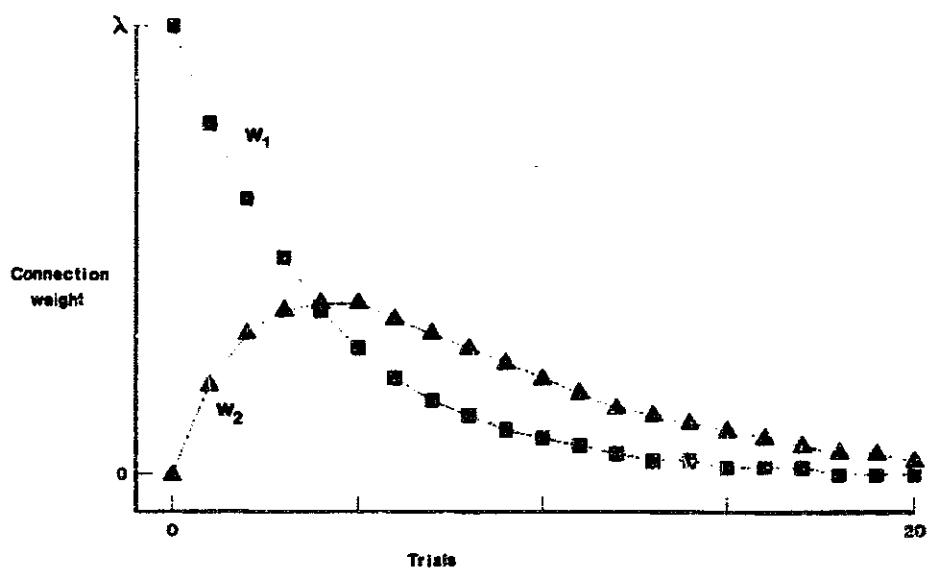


FIGURE 4.18. Connection weight values at the end of each trial in a simulation of higher order classical conditioning. CS_1 has been paired with a UCS until the weight w_1 reached the asymptotic value λ . For the trials shown, CS_2 and CS_1 are sequentially presented in the absence of the UCS causing w_2 to increase as CS_1 acts as a reinforcing stimulus for CS_2 . Since CS_1 is not being followed by the UCS, w_1 decreases to zero causing a similar decrease in w_2 .

4.5 Adaptive System Theory

In this section we discuss how the model we have presented is related to a variety of other learning rules used in adaptive system research. This will serve to place the model within a theoretical framework and indicate how it differs from learning rules proposed in the past. The history of adaptive systems research is too long and too diverse to exhaustively review here. Useful reviews are provided by Minsky (1963), Minsky and Selfridge (1961), Hawkins (1961), Holland (1975), and Klopf (1979, 1981). Even by restricting attention to adaptive systems based on "neural" mechanisms, we would be unable to give more than a cursory treatment. Arbib, Kilmer, and Spinelli (1976) provide a good, though also non-exhaustive, review of adaptive neural models. Here we focus only on rules that have received the most attention and are most closely related to our model.

Consider a generalized learning rule (as in Amari, 1977a): A synaptic weight increases or decreases in proportion to a reinforcement signal r :

$$w_i(t + 1) = w_i(t) + c r_i(t) \quad (4.8)$$

where c is a positive learning rate constant, $w_i(t)$ is the weight of synapse i at time t , and $r_i(t)$ is the

reinforcement signal to synapse i at time t . We are using the term "reinforcement signal" simply to denote that signal which determines the changes in connection weights. For some of the learning rules this signal only vaguely resembles what would be called reinforcement in behavioral studies.

4.5.1 Hebbian Rule

Within this framework the Hebbian postulate, in the form which we briefly discussed above, is formulated by letting $r_i(t) = x_i(t)y(t)$ in Equation 4.8. The most highly developed application of the Hebbian learning rule is its use in networks which implement associative information storage (e.g., Amari, 1977a, b; Anderson et al., 1977; Kohonen, 1977; Nakano, 1972; Wigstrom, 1973). The network shown in Figure 4.19 transforms stimulus patterns $X = (x_1, \dots, x_n)$ to response patterns $Y = (y_1, \dots, y_m)$. The inputs x_i act on the elements in exactly the same manner as the inputs x_i , but are used to specify patterns $Z = (z_1, \dots, z_n)$ to be associated with the stimulus patterns X . Repeated presentations of k different pairings of stimulus patterns $(x_1, z_1), \dots, (x_k, z_k)$, causes the network to learn, using the Hebbian learning rule, to elicit z_a when presented with x_a alone, $a = 1, \dots, k$. This occurs provided the patterns x_1, \dots, x_k form an orthogonal set.

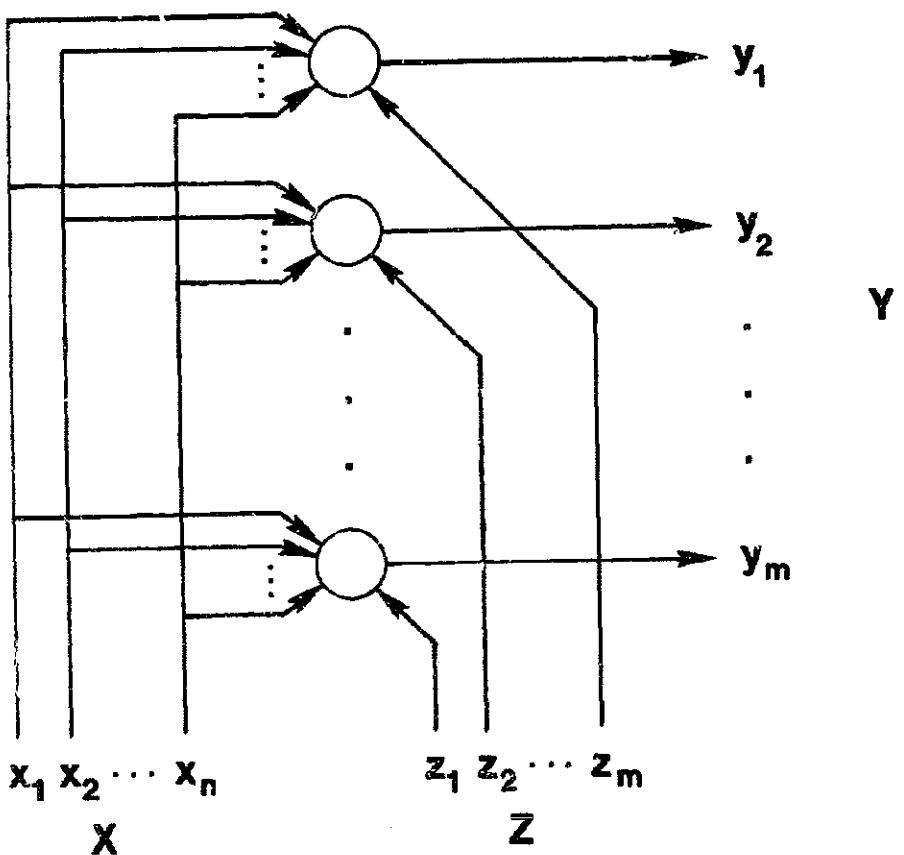


FIGURE 4.19. An associative memory network consists of a bank of m adaptive elements sharing the same n input pathways. Any of the many types of adaptive elements proposed can be studied in this configuration. Although each type of element leads to different storage and retrieval capabilities, all such networks show the properties of generalization, noise resistance, and content addressability which have stimulated interest in these structures.

The matrix of synaptic weights (w_{ij}) converges to the correlation matrix of the patterns X_α and Z_α , $\alpha = 1, \dots, k$ [footnote].

What accounts most strongly for the current widespread interest in associative memory networks is that they exhibit properties suggestive of the aspects of memory emphasized by Gestalt or mass action theorists (e.g., Freeman, 1975, and John and Schwartz, 1978). Since information can be stored in distributed form, associative performance may not be seriously impaired by various kinds of "lesions" (e.g., Wood, 1978). Distributed storage also provides for interesting forms of generalization and content addressability (e.g., Kohonen, 1977; Nakano, 1972, and Wigstrom, 1973).

These models provide evidence that learning rules which are essentially connectionistic in character need not imply a locationalistic view of memory. The theory of associative memory networks is well understood, and as research continues on mechanisms of this type, the result emerging is

It actually converges to the correlation matrix, also called the covariance matrix, only if the averages of the input patterns X_α and Z_α are zero. Amari (1977a) shows how the Hebbian rule can be modified to remove this restriction.

that any application of simultaneous, or spatial, correlation can be cast in a form that a Hebbian rule can implement.

However, as we indicated when discussing temporal relationships, the temporal subtleties of classical conditioning are not produced by the Hebbian rule even with the use of delays and other modifications. One would therefore not expect the processing capabilities of networks of Hebbian adaptive elements to extend far beyond spatial correlation.

4.5.2 Widrow-Hoff Rule

For the Widrow-Hoff rule the reinforcement signal is defined as follows:

$$r_i(t) = [z(t) - y(t)]x_i(t) \quad (4.9)$$

where

$$\text{and } y(t) = \sum_{j=1}^n w_j(t)x_j(t), \quad (4.10)$$

$z(t)$ and $x_i(t)$ are real numbers.

This rule requires the use of a specialized signal z which acts differently from the other input signals due to its special role in Equation 4.9 and the fact that it does not participate at all in the computation of the output y given

by Equation 4.10. This rule causes the weights to converge so that the response is a particular desired real number for each stimulus pattern X_α , $\alpha = 1, \dots, k$, the value to be associated with it is presented as z , call it z_α , then after sufficient repetitions of the pairs (X_α, z_α) , the element will respond with z_α when presented with X_α alone, $\alpha = 1, \dots, k$. The rule implements an iterative algorithm for computing a solution to a set of linear equations. A solution exists if the stimulus patterns X_1, \dots, X_k are linearly independent.

If the stimulus patterns are not linearly independent, convergence can still occur if the rule is modified by making the learning rate parameter c a variable whose value approaches zero as the trials continue, e.g., $c(t) = c/t$. In this case, and provided the pairs (X_α, z_α) occur with sufficient frequency in the input sequence, the weights converge so as to minimize the sum of the squared error over the stimulus patterns; that is,

$$\sum_{\alpha=1}^k (y_\alpha - z_\alpha)^2$$

is minimized where y_α is the element's output for pattern X_α , and z_α is the desired output. In this form, the Widrow-Hoff rule is an iterative algorithm for forming the Moore-Penrose pseudo-inverse of a linear operator which is the same as saying that it computes a linear regression. Duda and Hart (1973) provide a useful discussion of this and

closely related stochastic approximation procedures. This rule was proposed in the form of an adaptive element by Widrow and Hoff (1960).

Amari (1977a, b) discusses associative memory networks of neuron-like elements which rely on the Widrow-Hoff rule to form associations. In discussing the associative network shown in Figure 4.19, we said that when Hebbian synapses are employed, perfect recall of z_α upon presentation of X_α , $\alpha = 1, \dots, k$, was possible only when the stimulus patterns formed an orthogonal set. Using the Widrow-Hoff rule, perfect recall occurs even if the stimulus pattern set is only linearly independent. Amari (1977a, b) calls this "orthogonal learning" since non-orthogonal patterns are "orthogonalized" by the network. For sets of stimulus patterns that are not linearly independent, recall of the best pattern in the least-mean-square sense can be achieved.

A fact that is not generally realized is that the Widrow-Hoff rule is essentially identical to the Rescorla-Wagner equation. To see this, identify t with the trial number, each input with a CS, and the z signal with the UCS so that $z = \lambda$ when the UCS is present and $z = 0$ otherwise. In the Rescorla-Wagner equation (Equation 4.7) the presence of a CS_i input signal on a trial is indicated by the set notation $i \in S$, while the Widrow-Hoff form uses

x_i nonzero to indicate input signal presence on a trial and $x_i = 0$ to denote absence. The relevant equations and correspondences are:

Rescorla-Wagner:

$$\Delta v_{CS_i} = \begin{cases} c[\lambda - \sum_{j \in S} v_{CS_j}] & \text{for } i \in S \\ 0 & \text{for } i \notin S \end{cases}$$

Widrow-Hoff:

$$\Delta w_i = c[z - \sum_{j=1}^n w_j x_j] x_i$$

Correspondences:

$$w_i = v_{CS_i},$$

$z = \lambda$ if the UCS is present, otherwise $z = 0$,

$x_i = 1$ if CS is present, otherwise $x_i = 0$.

That these two models are, in fact, identical is striking since they were constructed for very different purposes. The Widrow-Hoff rule was formulated as an algorithm to solve sets of linear equations, and its theory addresses convergence properties. Not only are stimulus context effects not discussed in this theory, their existence is entirely incidental. The Rescorla-Wagner theory was proposed to compactly describe a wide variety of effects observed in animal learning experiments. That it also provides an important algorithm with a strong

connection to very useful areas of applied mathematics is fortuitous. We feel that the confluence of mathematical and empirical facts represented by what we shall call the Rescorla-Wagner/Widrow-Hoff rule might have considerable significance for understanding associative learning.

Due to its similarity to the Rescorla-Wagner model, the Widrow-Hoff rule provides a more adequate model of classical conditioning than does the Hebbian model. Unlike the Hebbian model, however, it does not provide a simple explanation for a stimulus substitution view of conditioning. Figure 4.4 shows the Widrow-Hoff rule as a model of classical conditioning. The specialized input z corresponds to the UCS. Since z does not directly influence the element's output, the UCR and CR must use separate pathways (compare to Figure 4.3). Also unlike the Hebbian rule, the Rescorla-Wagner/Widrow-Hoff rule has the property that weight modifications can only be driven by the specialized "teacher" input z .

A learning rule closely related to the Rescorla-Wagner/Widrow-Hoff rule is the perceptron rule of Rosenblatt (1962). If $z(t)$ in Equation 4.9 is restricted to taking only the values 0 and 1 and the output is similarly restricted by the use of a threshold, Equation 4.9 gives the fixed increment perceptron rule. This rule is an iterative

procedure for solving a set of linear inequalities. A solution exists if the desired response is a linearly separable function of the stimulus patterns. Nilsson (1965) provides several proofs of convergence, and Minsky and Papert (1969) discuss its limitations as a pattern recognition system. Despite these limitations, the perceptron learning rule has resurfaced, in slightly disguised form, as a way of storing data in associative memory structures (Albus, 1979; Amari, 1977a, b). These applications illustrate that in certain applications, and using certain ways of representing data, the limitations of linear learning rules are not as devastating as once thought.

The perceptron seems to be most often thought of as a model of instrumental conditioning in which reinforcement is contingent on the response rather than one of classical conditioning which involves no response contingencies. This view, however, is mistaken. If the error signal $z(t) - y(t)$ in Equation 4.9 is taken as being computed by the perceptron's environment, then the perceptron can be viewed as a response contingent system: If the response is correct, the error is 0; if it is incorrect, the error is 1 or -1. However, this feedback through the environment is of such stereotyped form that it can be eliminated, for arbitrary environments, by just letting the error be

computed by the perceptron itself with the environment always simply providing the desired response rather than an error signal. Viewed in this manner, the perceptron is essentially the same as the Rescorla-Wagner model: It compares its own response (expectation) with the correct one (UCS) and modifies the weights in order to make them agree. The instrumental conditioning paradigm, on the other hand, involves essential feedback through the organism's environment; that is, feedback which cannot be eliminated in a uniform way for all environments. Nontrivial forms of response-contingent learning have received very little attention by adaptive network theorists.

4.5.3 Rescorla-Wagner/Widrow-Hoff Predictor

The Rescorla-Wagner/Widrow-Hoff rule does not produce the predictive aspect of classical conditioning. Here we discuss the minimal modifications to that rule which enable it to produce predictive or anticipatory responses. From the resulting rule, which we call the Rescorla-Wagner/Widrow-Hoff predictor, it is possible to see what additional properties our model provides. While we know of no instance in which the Rescorla-Wagner/Widrow-Hoff predictor is used in an adaptive network theory, it is an example of a linear prediction procedure and is part of a larger theory of prediction or forecasting (see, for

example, Box and Jenkins, 1976).

For the Rescorla-Wagner/Widrow-Hoff predictor the reinforcement signal is defined as follows:

$$r_i(t) = c[z(t) - y(t - \tau)]x_i(t - \tau) \quad (4.11)$$

where $y(t)$ is as defined by Equation 4.10 and τ is some positive constant. Changes in connection weights are such as to reduce the difference between $z(t)$ and $y(t - \tau)$ so that an equilibrium is approached at which $z(t) = y(t - \tau)$, or $z(t + \tau) = y(t)$. This means that the element will learn to produce activity that anticipates by τ the activity of the UCS pathway z if the input contains enough predictive information. More precisely, recalling the discussion of the Widrow-Hoff rule above, if c is allowed to decrease as conditioning proceeds, this element will produce a best least squares prediction by of the signal z . All of the stimulus context effects of the Rescorla-Wagner/Widrow-Hoff rule are also produced by the predictor.

The process defined by the predictor can be described as follows: Activity on an input pathway possibly causes a response but also causes the connection from that pathway to become eligible for modification a certain period of time (τ) later. An eligible connection is modified only if the UCS signal strength differs from the expected strength.

Thus, each time $z(t)$ deviates from $y(t - \tau)$ the input pathways that were active earlier (and thus are eligible) will modify their connection weights, or associative strengths, w_j . The reinforcement signal is a measure of how strongly the current UCS confirms or contradicts the previously formed expectation or prediction.

As a model of classical conditioning the predictor defined by Equation 4.11 requires an ISI exactly equal to τ for any conditioning to occur. This limitation can be eliminated, along with the arbitrariness of the choice of τ , by replacing the delayed signals $x(t - \tau)$ and $y(t - \tau)$ in Equation 4.11 by more general forms of traces such as those used in our model. Let $\bar{x}_j(t)$ and $\bar{y}(t)$ be some weighted averages of their respective function values over some time interval preceding t produced using Equations 4.3 and 4.4. Then the reinforcement signal for the Rescorla-Wagner/Widrow-Hoff predictor becomes:

$$r_j(t) = c[z(t) - \bar{y}(t)]\bar{x}_j(t). \quad (4.12)$$

The temporal relationships implied by this rule depend on the characteristics of the CS and UCS, the form of the traces \bar{x}_j and \bar{y} , and the parameters of the experimental paradigm. Some details of these dependencies are presented in Appendix A.

4.5.4 Uttley's Informon

Uttley (1970, 1975, 1976a, b, c, 1979) has suggested a learning rule which is closely related to the Rescorla-Wagner/Widrow-Hoff procedure except that it conforms to some of the constraints of the Hebbian rule. Starting with the Widrow-Hoff rule (Equation 4.9), let $z(t) = -w_0x_0(t)$ where w_0 is a fixed positive number. That is, let the specialized "teacher" input be a signal to a fixed inhibitory pathway. It is further assumed that this fixed signal participates in the computation of the output y just like any other input signal, then Equation 4.9 can be rewritten as follows:

$$r_i(t) = [-w_0x_0(t) - \sum_{j=1}^n w_j(t)x_j(t)]x_i(t) \quad (4.13)$$

$$\begin{aligned} &= -[\sum_{j=0}^n w_j(t)x_j(t)]x_i(t) \\ &= -y(t)x_i(t). \end{aligned}$$

This is the Hebbian rule except for the minus sign. Uttley argues that this change of sign is desirable since it changes the positive feedback inherent in the Hebbian rule to negative feedback desirable for its stabilizing influence. Coincidence of pre- and postsynaptic discharges decreases rather than increases synaptic strength. The equilibrium weight values are those which result in zero total input to the element. Uttley notes the similarity of

this rule to the Rescorla-Wagner model and illustrates how it can produce much of the same behavior (Uttley, 1975).

Uttley describes his model in the above manner but actually simulates a more complex model based on the concept of "mutual information." He uses exponentially weighted time averages to estimate the negative of the mutual information between input and output signals. At each time step, the weights are set to these estimates. Although the concept of mutual information led Uttley to the informon model and provides an interesting view of the stimulus contingencies which produce learning, it is an unnecessary complication to what is essentially the Widrow-Hoff rule.

If the special input labelled z in Figure 4.4 is regarded as a fixed inhibitory input, then that figure shows the use of Uttley's element in an analog of the classical conditioning paradigm. This is identical to the corresponding situation or the Rescorla-Wagner/ Widrow-Hoff and perceptron models. Here, however, there is the additional consequence that the UCS actually inhibits the CR both before and after learning. This is due to the treatment of the UCS as an inhibitory signal that is used in the computation of the element's output. In the Rescorla-Wagner/Widrow-Hoff and perceptron models, the UCS is a special input that never influences the output of the

element except indirectly through the learning process.

In order to obtain the stability and stimulus context effects of the Rescorla-Wagner/Widrow-Hoff rule, while at the same time adhering to the basic constraints of the Hebbian rule, Uttley had to abandon the simple stimulus substitution view of conditioning provided by the Hebbian rule and make the behaviorally unsupportable assumption that the UCS inhibits the CR.

By retaining the form of the Hebbian rule, however, the informon has the property that even though there are fixed, prespecified classifying input channels, these channels are not the only sources of signals which can cause weight modifications. This is an important property, but it can be obtained in an entirely different manner (as illustrated by our model) which also has the advantages of the Rescorla-Wagner/Widrow-Hoff rule, but retains a stimulus substitution view, produces appropriate ISI dependency, and permits the CR to begin before the UCS. While we feel that Uttley's approach represents an independent discovery of the advantages of the Rescorla-Wagner/Widrow-Hoff rule, we also feel that it needlessly adheres too closely to the original Hebbian postulate.

4.5.5 Our Model

Within the framework provided by Equation 4.8, our model uses a reinforcement signal defined as follows:

$$r_i(t) = [y(t) - \bar{y}(t)]\bar{x}_i(t) \quad (4.14)$$

where y is as defined by Equation 4.10 and \bar{y} and \bar{x}_i are traces of their respective signals as described above. This differs from the Rescorla-Wagner/Widrow-Hoff predictor (Equation 4.12) by the substitution of $y(t)$ for the specialized reinforcing signal $z(t)$. This eliminates the requirement for reinforcement to be provided only by a fixed reinforcing pathway. Since $y(t)$ can be affected by activity on any input pathway, any input signal can bring about changes in the efficacies of other pathways. This permits the adaptive element to extract predictive relationships among its inputs in the same way that a Hebbian element or an informon extracts simultaneous associations. Unlike the informon, however, our model retains the stimulus substitution properties of the Hebbian model since the CR and the UCR share the same pathway.

We have been able to eliminate the need for a distinct channel for reinforcing signals by, in effect, providing a distinct time (with respect to a CS) for reinforcement. This was suggested by work of Klopff (1972, 1981) in which a

similar method was proposed for eliminating the requirement that response-contingent reinforcement be delivered over a specialized channel. Here we have restricted this idea to classical conditioning.

We note that it is possible to use our model in an associative memory system such as those described above which rely on the Hebbian rule or the Rescorla-Wagner/Widrow-Hoff rule. One would obtain a network capable of exhibiting the properties of our model together with the properties of distributed, associative information storage. We have not yet systematically explored the implications of such a system, but it is unlikely that it would lack any of the properties which have stimulated interest in this kind of associative memory structure. In particular, such a system would show that our model, although connectionistic in character, need not imply a locationalistic theory of memory.

4.6 Stability and Saturation

Some issues that were not directly addressed in the preceding section concern technical problems that occur when networks of elements based on various learning rules are simulated. For example, a literal application of the Hebb postulate implies a positive feedback loop (increases in

excitatory synaptic weights cause higher correlation between pre- and postsynaptic activity and hence further weight increases). Excitatory synaptic weights tend to become large irrespective of the significance of the input signals, and some additional mechanism is required to prevent the strengths of all connections from growing without bound or from reaching and remaining at their maximum values. Early computer simulations illustrated the importance of solving these problems for preventing network "seizures" (Rochester, Holland, Haibt, and Duda, 1956).

Here we discuss several approaches to solving the stability and saturation problems associated with the Hebb rule and relate them to the solution provided by the learning rule discussed here. Our point is to show that learning rules which are based on the Rescorla-Wagner/Widrow-Hoff rule, such as ours and that proposed by Uttley, not only provide more valid models of classical conditioning than the Hebbian rule, but also solve these technical problems in a simple way. While there is no logical or empirical necessity that the stimulus context effects accounted for by the Rescorla-Wagner/Widrow-Hoff rule arise at a cellular level, it is suggestive that if they did, then additional mechanisms would not be required in order to solve stability and saturation problems.

Some of the current approaches to solving these problems (notably Grossberg's, e.g., 1969, 1974, 1976a, b) stress the importance of careful network design and use of inhibitory connections for controlling network stability. Other approaches attempt to achieve similar results by modifying the original Hebbian postulate so as to incorporate local stabilizing mechanisms which operate irrespective of an element's network environment. This latter approach has not been shown to be sufficient for solving network stability problems but does contribute to their solution by making the adaptive changes inherently more manageable. While we feel that network level considerations (i.e., *a priori* structure) are very important, they are strongly influenced by the choice of local learning rules, and here we focus only on element level issues.

There are two fundamentally different ways of preventing unbounded weight growth in theoretical models of plastic synapses. The first technique is to impose an upper bound at some fixed, pre-determined value. Whether this is done by setting the weight back to the preset maximum whenever an increment makes it larger, or whether the learning rule is modified in order to make the value asymptotically approach a preset finite limit, the same problem arises: Unless weights decrease in some

circumstances, all excitatory weights will tend to reach and remain at their maximum values. Saturation of some weights may be desirable, but if all the weights always eventually reach their maximum value, then all learning eventually ceases, and all stored information is eventually forgotten. Either learning must occur slowly enough to postpone this ultimate state of forgetfulness for as long as necessary, the plasticity of some connections must be temporary, or a means for decreasing weights must be introduced.

The second technique for preventing unbounded weight growth relies on the boundedness of the reinforcement signal which drives the weight modification process. Instead of a fixed, predetermined limit being enforced by the learning rule, the limit is a function of the external reinforcing input to the element. Larger reinforcement can always cause weights to increase but, and this is the crucial point, arbitrarily prolonged periods of nonzero reinforcement must not produce arbitrarily large weights. Several of the methods discussed below solve stability and weight saturation problems in this manner.

4.6.1 Normalization

One of the most common techniques used in simulations invokes a "conservation of total synaptic strength", or

normalization, principle. This technique is a particular way of presetting weight bounds. The total saturation problem is avoided by requiring some weights to decrease in order to maintain the sum of all the weights at a constant value. New weight values w_i' are computed according to the Hebbian rule, and then each w_i' is divided by the sum of all of the w_i' to obtain the actual next weight values, that is (cf. von der Malsburg, 1973),

$$w_i(t+1) = w_i' / \sum_{j=1}^n w_j' \quad (4.15)$$

where

$$w_i' = w_i(t) + c x_i(t) y(t). \quad (4.16)$$

This normalization procedure is successful in permitting those pathways to dominate whose activity is most strongly correlated with postsynaptic activity. One can view synaptic strength modification computed in this way as a competition among pathways for proportions of the sum. In many models, this procedure is absolutely essential, not only for stable operation of the model, but also for the generation of behavior which resembles experimental data (e.g., von der Malsburg, 1973).

While adequately solving some of the technical problems associated with the Hebbian rule, this normalization procedure has several deficiencies. First, it was pointed out by Uttley (1976a) that although perfect normalization

often produces desired results, small departures from this ideal can cause rather drastic changes in behavior. If, for example, one synapse is consistently favored by even a very small amount in the normalization process, then it can gain much more than its share of the total synaptic strength. The weight values can reflect normalization asymmetries rather than the desired correlation measures.

A second criticism of the normalization procedure holds to the extent that a faithful representation of classical conditioning phenomena is desired. Although stimulus context effects are produced by normalization since this technique does cause weights to change in a manner dependent on all the other weights, these context effects are different from those observed experimentally. For example, suppose each input x_i , $i = 1, \dots, n$, to an adaptive element using a normalized Hebbian scheme has a constant value $x_i(t) = x_i$ for all t . Using Equations 4.15 and 4.16, it is not hard to show that the equilibrium weights are

$$w_i = x_i / \sum_{j=1}^n x_j, \quad i = 1, \dots, n. \quad (4.17)$$

These equilibrium values are independent of the initial weight values. The stimulus context effects observed experimentally, however, require that the associative strengths at the beginning of a series of trials crucially determine their values at its end.

Consider blocking, for example. Suppose an element has two binary valued inputs x_1 and x_2 corresponding to conditioned stimuli CS_1 and CS_2 and an input of arbitrary fixed strength representing an unconditioned stimulus. Assume that the associative strengths w_1 and w_2 of CS_1 and CS_2 initially equal zero and are thereafter required always to sum to one. Pairing CS_1 with the UCS until equilibrium is reached results by Equation 4.17 in $w_1 = 1$ and $w_2 = 0$. Now, starting with these values and pairing both CS_1 and CS_2 with the UCS results by Equation 4.17 in equilibrium values $w_1 = w_2 = .5$. This is the same result that would have been produced if the weights were both zero at the commencement of the paired trials. Blocking, on the other hand, would occur if the series of paired trials did not change the weights from the values they had when it began; that is, $w_1 = 1$, $w_2 = 0$.

Another criticism of the normalization technique can be made if a model using this method is intended to reflect what might occur in actual neurons. Although it has been suggested that synaptic modifications and normalization may be the result of the redistribution of a constant amount of receptor protein (Stent, 1973), this hypothesis goes far beyond available data given the lack of corroborative support from other lines of evidence. One way of meeting the criticism that normalization is an unlikely cellular

mechanism is to postulate that normalization occurs at a network rather than at a cellular level. The work of Grossberg and his colleagues (Ellias and Grossberg, 1975; Grossberg, 1974, 1976a, b) exemplifies this approach.

4.6.2 Autonomous Decay

If it is assumed that synaptic strength slowly decays in the absence of a reinforcement signal, then a bound on weight size is imposed that is a function of reinforcement level and the decay rate. A weight can always increase when its reinforcement signal increases, but if the reinforcement signal remains bounded, then no matter how long the signal persists the weight also remains bounded. Thus, learning can occur whatever the system's "age," but experiences are always "forgotten" within a certain period of time. In system theoretic terms, the adaptive element has "definite memory": It cannot remember anything that occurred arbitrarily far in the past. Moreover, the weight bound is inversely proportional to the length of time that memory traces can be retained. That is, if weights are to be kept below rather low levels, then the decay of the weights must be rather fast. The normalization method described above, in contrast, has "indefinite memory," meaning that information is not lost unless it is actively replaced by new information.

Despite the lack of indefinite memory, learning rules incorporating autonomous decay lead to behavior that can be understood in mathematical terms. For example, if the decay is sufficiently slow, then the longterm statistical properties of the reinforcement signal can be reflected in the weight values (e.g., Amari, 1977a, b; Uttley, 1976a, b, c).

4.6.3 Negative Feedback

While normalization and autonomous decay schemes employ negative feedback, rules resembling the Rescorla-Wagner/Widrow-Hoff procedure use a more explicit form. Uttley (1976a) directly eliminates the positive feedback inherent in the Hebbian rule by changing it into negative feedback by reversing the sign (Equation 4.13). Coincidence of pre- and postsynaptic discharges decreases rather than increases the synaptic strength. This produces the stimulus context effects observed in classical conditioning experiments while at the same time solving stability and saturation problems (and producing the undesirable consequences discussed earlier). However, it is not the precise form of Uttley's model which produces these solutions, but rather the model's resemblance to the Rescorla-Wagner/Widrow-Hoff rule.

Recall that for the Rescorla-Wagner/Widrow-Hoff rule weights change according to

$$\Delta w_i(t) = c[z(t) - \sum_{j=1}^n w_j(t)x_j(t)]x_i(t).$$

A weight therefore cannot change if either the input signal on that pathway is zero ($x_i(t) = 0$) or the total stimulus strength equals the training signal $z(t)$. Thus, the weights are always bounded, yet never saturate so as to be insensitive to further changes in the environment. Negative feedback is provided in the form of the expectation term

$$\sum_{j=1}^n w_j(t)x_j(t).$$

Learning can always occur when the reinforcement differs from the expected level, and the asymptotic weight values depend on the magnitude of the reinforcement signal. Moreover, the rule permits memory of events which occurred arbitrarily far in the past; that is, it has indefinite memory [footnote]. A weight will not decrease unless a

It is curious that the model Uttley actually simulates does not possess indefinite memory. Exponentially weighted time averages are used to estimate the negative of the mutual information between input and output signals. This makes the informon's stability an obvious property, but memory traces always decay to zero due to the exponential decay used to estimate mutual information (Uttley, 1970). If no estimates of mutual information were used, then indefinite memory would be present without additional mechanisms, and none of the rule's other advantages would be lost.

stimulus occurs that is not reinforced to the expected level.

The model we have presented uses negative feedback in the form of an expectation term which is a weighted average of past values of the element's output. The weight associated with a pathway cannot change unless that pathway is eligible and the current output value differs from the weighted average of past output values. As a weight grows, the signals arriving on that pathway cause larger output values and hence larger expectations. As the expectations grow, they exert negative effects on weight growth. The stability of this method is evident from the simulation results shown above.

One consequence of this form of negative feedback is that if a signal arrives via a modifiable pathway, for example, as a positive rectangular pulse, but is not followed by other activity within the eligibility period, then the weight of that pathway will, if positive, decrease toward zero. This will occur since the signal offset will coincide with positive eligibility to cause a negative change in weight. With repeated presentations of stimuli not directly followed by other activity, weights will converge to zero. This is why in the simulations of classical conditioning presented above we required the UCS

to arrive over a pathway of fixed weight. If this weight were not fixed, then UCS presentation would eventually have no effect on the output of the adaptive element. This does not imply, however, that our model has definite memory. In the absence of incoming signals, a pathway will exhibit no change (since it will never be eligible for modification) no matter how long the period of inactivity lasts.

It is useful to compare the form of negative feedback employed by models resembling the Rescorla-Wagner/Widrow-Hoff rule with that of normalization schemes. In the former case, the feedback signal is the total stimulus strength, while in the latter it is simply the sum of the weights. Although one form of feedback is additive and the other is multiplicative, the major difference is that the Rescorla-Wagner/Widrow-Hoff rule uses information from the current stimulus pattern while the normalization scheme does not. Without this information, the stimulus context effects observed experimentally cannot be produced. Further, if one is arguing for the cellular plausibility of a learning rule, then negative feedback in the form of total stimulus strength is easier to account for than feedback in the form of total synaptic efficacy. Since total stimulus strength is reflected in neurons by the membrane potential, it is plausible to hypothesize that this signal is available, at least approximately, at each

synaptic site.

We have seen, then, that not only does the Rescorla-Wagner/Widrow-Hoff rule provide a strong model of classical conditioning and a powerful iterative method for solving sets of linear equations, it also solves some of the technical problems which always accompany the use of the Hebbian rule. Our model retains these advantages while accounting for some of the intratrial temporal structure of classical conditioning.

4.7 Cellular Mechanisms

There is always a risk in speculating about cellular mechanisms for learning processes. On the one hand, not enough is known about the cellular changes which occur during associative learning to permit the construction of detailed models. On the other hand, experimental progress in this area is occurring so rapidly that any postulated mechanism is likely to be soon invalidated by concrete fact. Despite these hazards, we feel that a discussion of our model in light of current electrophysiological and biochemical knowledge of cellular plasticity can be of value since the model is empirically supported at a behavioral level and is of interest from a theoretical perspective. In addition, the concepts of "eligibility" and "expectation" in

our model are not only of critical importance in accounting for animal learning behavior and, we believe, essential for adaptive behavior of artificial systems, but can be associated quite naturally, albeit speculatively, with certain processing capabilities of neurons.

There are four aspects of our rule to consider. First, the notion of eligibility would be realized if a synapse were "tagged" by a non-stimulating trace for some period of time after each discharge of the presynaptic cell. This indication of previous stimulation would be required to endure for a period on the order of at least a few seconds rather than a few milliseconds. This trace should remain local to the synapse. Second, some way of registering changes in the postsynaptic cell's firing rate from its previous level is required. This determines the reinforcement which facilitates or inhibits eligible synapses. The length of time over which the reference firing rate is determined is not critical but should be relatively long, perhaps with a time scale similar to that of eligibility. Third, it is necessary for the measure of eligibility, which is local to the synapse, to interact with the reinforcement signal, which is a global feature of the postsynaptic cell. This interaction should occur at each synapse. Finally, the result of the interaction between the eligibility of a synapse and the reinforcement level must

regulate modifications of the transmission efficacy of that synapse.

The notion of a synaptic marker indicating previous presynaptic discharges could be realized either postsynaptically or presynaptically. We discuss a postsynaptic site for eligibility first. There is good evidence that in some cells the binding of a neurotransmitter to its receptor site regulates postsynaptic concentrations of an adenosine 3', 5'-monophosphate (cyclic AMP) or guanosine 3', 5'-monophosphate (cyclic GMP). It has been hypothesized that these cyclic nucleotides may mediate, as second messengers, the action of several neurotransmitters in generating slow postsynaptic potentials. This hypothesis is supported in several preparations by several lines of electrophysiological and pharmacological evidence. For reviews see Greengard (1976), Nathanson (1977), or Rasmussen, Jensen, Lake, Friedmann, and Goodman (1975).

However, studies of other preparations have suggested that postsynaptic increases in cyclic nucleotide concentrations may have roles other than the generation of postsynaptic potentials. For example, it has been shown that the administration of cyclic AMP and cyclic GMP to cells in a sympathetic ganglion of the bullfrog does not

cause appreciable changes in membrane potential even though synaptic stimulation increases both cyclic AMP and cyclic GMP in these cells (Busis, Weight, and Smith, 1978). It has been suggested that in addition to the role cyclic nucleotides may play in simple neurotransmission, they may also carry more indirect messages which might, for example, mediate a stimulus trace which temporally links events in associative learning at a cellular molecular level (Woody, 1976).

Although the role of cyclic nucleotides in synaptic transmission and its regulation is not yet clear, it is evident that in some cells, and for some neurotransmitters, postsynaptic concentrations of cyclic nucleotides do reflect the amount of presynaptic stimulation received and can register previous stimulation for a time which is very long compared to the millisecond times of electrical activity. A difficulty, however, with the hypothesis that postsynaptic chemical concentrations provide stimulus traces as required by our model is that these traces would probably not remain local to their initiating synapses.

The locality of the trace suggests that a presynaptic site might be more plausible. Studies of the presynaptic mechanisms which are responsible for the nonassociative synaptic changes of post-tetanic facilitation and

habituation suggest that the notion of eligibility could be represented presynaptically. Since these nonassociative instances of synaptic modifiability involve time scales much longer than that of electrical activity, we might postulate that some of the same mechanisms realize the notion of eligibility used in our model of classical conditioning. For example, intracellular concentration of free Ca^{2+} or Ca^{2+} conductance characteristics (e.g., voltage dependence) could provide relatively prolonged records of presynaptic activity. The mechanisms which result in post-tetanic facilitation or habituation for some temporal stimulus patterns might provide important record keeping facilities which operate whatever the stimulus characteristics are.

If eligibility were recorded presynaptically, then we would need to postulate some way in which the activity of the postsynaptic cell could influence the presynaptic terminal. Although it has been shown that postsynaptic activity can influence a presynaptic terminal by altering the ionic content of the surrounding medium (Weight and Erulkar, 1976), we discuss instead mechanisms whose roles in synaptic modulation are much better understood. These involve presynaptic facilitation via synapto-synaptic connections. Figure 4.20 shows a simple circuit in which presynaptic modulation is provided by extracellular feedback from the postsynaptic cell. The figure shows a single

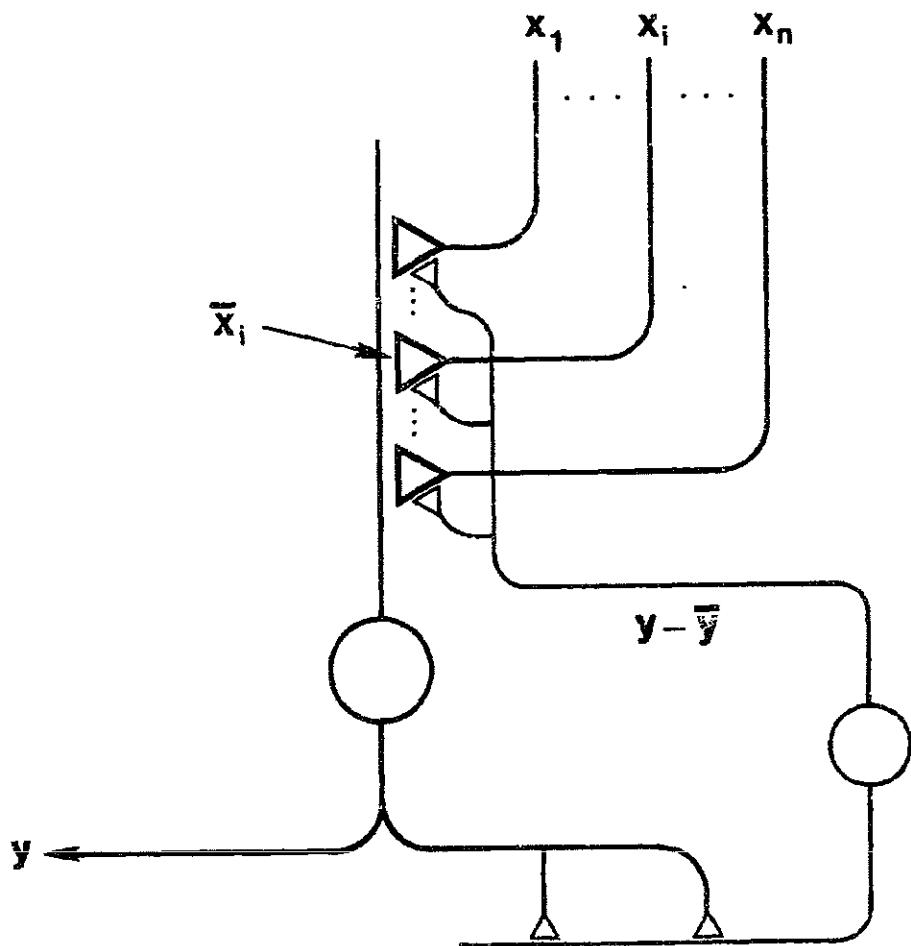


FIGURE 4.20. The adaptive element implemented via a feedback interneuron. Eligibility \bar{x}_i is computed presynaptically, and the difference between actual and expected firing rates, computed by the feedback interneuron, modulates synaptic strengths through synapto-synaptic connections.

feedback interneuron, but a multisynaptic pathway is clearly also possible. In fact, the feedback pathway could pass through a brain region which integrates the signal with other information in a manner not accounted for by our model. For example, the signal may be integrated with other stimulus context information by the septo-hippocampal complex in a way similar to that suggested by Moore (1979). Figure 4.20 also shows episynaptic connections from the interneuron to all of the incoming fibers. As formulated here, our model requires this feature, but it should be regarded as a convenient simplification. Fibers not contacted by the interneuron would not exhibit plasticity (or, at least, not plasticity of the same form), and episynaptic connections carrying signals from other than the postsynaptic cell would permit processing of a form more complex than that considered here.

It remains to suggest how the feedback signal from the interneuron could represent the reinforcement signal, that is, the deviation of the postsynaptic cell's firing rate from previous levels. Perhaps the simplest possibility is that the interneuron, or the network of interneurons, responds only to changes in its input as has been commonly observed for some cells responding to sensory stimuli. A more complex hypothesis would be for the presynaptic input from the feedback interneuron(s) to produce two superimposed

effects on each synapse which it modulates. One effect would be a fast change in the voltage dependence of Ca^{2+} conductance so that depolarization would cause increased Ca^{2+} influx. The amplitude of this effect would depend on the eligibility of the modulated terminal determined by its previous history of depolarization. This increased influx of Ca^{2+} would facilitate the transmission effectiveness of the synapse by increasing transmitter release. The second effect would be a slower and less dramatic decrease in the peak Ca^{2+} conductance during depolarization. Again the magnitude of this effect would depend on the terminal's eligibility. This second effect would decrease transmitter release. If one assumes that these two effects linearly superimpose and that the effects of different discharges of the presynaptic terminal superimpose, then the resultant change in synaptic efficacy would depend on eligibility and the amount of change in activity of the feedback pathway as required by our model. Summing the fast positive and slow negative effects would produce a form of differentiation. The "expectation" would be represented by the negative component of the presynaptic effect.

We summarize this discussion of cellular mechanisms by making several observations. The model of classical conditioning which we have presented was formulated on the

basis of empirical evidence from behavioral experiments and from a sensitivity to the technical difficulties which have beset theoretical adaptive network studies. While the evidence that this model, or some variant of it, might be implemented at the level of single neural units or simple neural circuits is not strong, there is evidence that a relatively long lasting and non-stimulating memory of previous activity as is required by our notion of eligibility is indeed present at a cellular level. The monosynaptic phenomena of post-tetanic facilitation and habituation show that synapses themselves do possess nontrivial forms of short term memory which do not require one to hypothesize that reverberatory electrical activity stores reflections of previous activity. Other evidence exists suggesting that within a single cell can exist mechanisms for short term stimulus traces as well as longer term memory (e.g., Alkon, 1979; Libet, Kohayashi, and Tanaka, 1975; von Baumgarten, 1970; Weight, Schulman, Smith, and Busic, 1979; Woody, Carpenter, Gruen, Knispel, Crow, and Black-Cleworth, 1974).

4.8 Summary and Conclusions

While the spirit of Hebb's theory still seems to be relevant, there is little support for the use of a literal interpretation of the Hebbian rule in adaptive network

studies. As a model of classical conditioning, it is not up to the standard of sophistication now available in the learning theory literature. As a model of neural plasticity, it lacks experimental support and is based on a view of the processing capabilities of neurons and synapses which does not take into account the wealth of data now available. While networks employing Hebbian-style rules have been successful in producing some interesting effects, their behavior is far from the level of sophistication required for complex tasks. Finally, models relying on Hebbian-style rules require rather ad hoc additional mechanisms to insure stable and flexible behavior.

The Rescorla-Wagner/Widrow-Hoff rule, to which the perceptron and Uttley's informon are closely related, provides a more valid model of classical conditioning by incorporating stimulus context effects while at the same time cleanly solving a number of stability and saturation problems. That the Rescorla-Wagner equation was developed to account for animal learning behavior, while the nearly identical Widrow-Hoff rule was formulated to approximate the solutions of sets of linear equations, suggests that these rules describe some ingredient essential for adaptive behavior. One important aspect of the Rescorla-Wagner/Widrow-Hoff rule's behavior is the extraction of reliable and nonredundant information which

correlates with reinforcement. The experimental results regarding stimulus context effects in classical conditioning indicate that animals similarly form reliable and nonredundant associations.

We have presented a rule that preserves the properties of the Rescorla-Wagner/Widrow-Hoff rule but also incorporates the predictive nature of classical conditioning. The problems of making useful and accurate predictions seem to be solved by the ability to generate expectations. The actual events are then compared with those predicted, and appropriate incremental changes are made if the two differ. The Rescorla-Wagner equation does this while lumping together, as far as time of occurrence, all stimuli present on a trial. One contribution of the adaptive element developed here is to provide a mechanistic implementation of the descriptive Rescorla-Wagner theory of classical conditioning. In taking this lumped trial theory to a mechanistic form in which system behavior is specified at all times within the trial, it becomes possible to make distinctions between inputs based on their relative time of occurrence. Rather than extracting reliable and nonredundant information which correlates with reinforcement, this rule extracts reliable, nonredundant, and early predictors of reinforcement. Moreover, an adaptive element employing this rule is able to use its

sensitivity to predictive information to make predictions which occur earlier than the events predicted. A prediction made at the same time as, or later than, the event predicted is no more useful in guiding behavior than no prediction at all.

In addition, the adaptive element presented here preserves the simple account of stimulus substitution provided by the Hebbian rule. This is true since the UCR and CR share the same pathway--probably the simplest hypothesis accounting for the similarity of the UCR and the CR. Also, as in the case of the Hebbian model (and Uttley's informon), activity on any input pathway can cause changes in other pathways. This produces some higher order learning effects and permits the element to extract regularities whose constituents have not been predetermined by a priori network structure. Unlike the Hebbian and informon models, however, it extracts spatio-temporal rather than just spatial regularities.

Although we feel that our model includes some of the aspects of classical conditioning which have adaptive significance, the model is not a completely valid model of classical conditioning and obviously does not go beyond this restricted learning paradigm. It does not, for example, include the effects of experience on stimulus salience. In

addition, like the original Rescorla-Wagner model, our model makes only ordinal predictions about behavioral data. Recent extensions of the Rescorla-Wagner model to deal with these shortcomings (Frey and Sears, 1978) can perhaps also be applied to our adaptive element model. Our theory also does not address stimulus representation problems. We have assumed that input signals arrive at an adaptive element on discrete pathways of fixed "meanings." In a more sophisticated model, these meanings would be changed by "upstream" circuits as, for example, might occur in configural learning where a compound stimulus is treated as a nonlinear combination of its parts. Our theory does not indicate how the adaptive mechanisms we have suggested can be extended to extract arbitrary nonlinear regularities.

The model presented here also does not address the issues arising from response-contingent reinforcement paradigms. Although the exact nature of the relationship between classical and instrumental conditioning remains elusive (e.g., Rescorla and Solomon, 1967), the attention given to temporal processing in our model makes extensions possible which incorporate response contingencies. Klopf's (1972, 1981) theory, which forms the basis of several aspects of the model discussed here, incorporates response contingencies, and Sutton (1978c) has extended this theory to a single process view of expectation in classical and

instrumental conditioning using an adaptive element closely related to that presented here. However, a thorough formal treatment of these issues is beyond the scope of the present paper.

Also beyond this paper's scope is a discussion of the kinds of behavior which can be expected from networks of adaptive elements like those proposed here. Can such a network perform sophisticated learning tasks? This question is central from a theoretical perspective and notoriously difficult to answer for any type of primitive component. Here we merely suggest that the predictive capabilities of the adaptive element presented here may permit adaptive networks to exhibit forms of behavior not yet obtained from network models. Our reason for believing this is that predictive capabilities permit response alternatives to be evaluated before overt action is taken (see Section 7).

The model we have developed need not be thought of as a neural model. It is supported at a behavioral level and has potentially significant theoretical implications. However, the search for neural analogs of behavioral conditioning continues to guide learning and memory research in the neurosciences. From our discussion of cellular mechanisms it is clear that while there is no shortage of machinery for implementing almost any learning model one might construct,

there is evidence indicating that some of the essential aspects of our model could be implemented in a natural manner. The stimulus trace required by the notion of eligibility could involve either presynaptic or postsynaptic biochemistry. Our definition of reinforcement as the difference between actual and expected output levels can be realized via fast excitatory and slow inhibitory effects.

At the very least our discussion of cellular mechanisms makes it clear that the concept of a neuron as a biological logic gate that still pervades much neural network theory is much too simple. Neurons and their synapses possess processing capabilities that can utilize relatively long term histories of pre- and postsynaptic activity. In the terms of system theory, they possess a rich internal state space which can support behavior requiring nontrivial forms of memory. Neural network theorists have focused largely on synaptic weights as a form of memory and have postulated only relatively simple rules for controlling these memory variables. Other forms of memory have generally been assumed to be metabolic and genetic, and, to a first approximation, not significantly implicated in computational behavior. Notably absent from the theoretical literature is a consideration of potentially powerful forms of synaptically local short term memory and their possible roles in synaptic modulation. The growing understanding of

the role of biochemical mechanisms in synaptic action indicates that there is considerable internal memory linking events that occur at intervals of seconds, minutes, hours, and days. Moreover, this processing interacts with physiological events that occur in milliseconds. It seems certain that these mechanisms are crucially involved in neural plasticity. The model we have proposed takes a step toward recognizing the theoretical importance of the first few links in this chain.

SECTION 5

ASSOCIATIVE SEARCH NETWORK: A REINFORCEMENT LEARNING ASSOCIATIVE MEMORY *

5.1 Introduction

Numerous reports have appeared in the literature describing associative memory systems in which information is distributed across large areas of the physical memory structure (e.g., Amari, 1977a, b; Anderson et al., 1977; Cooper, 1974; Kohonen, 1977; Nakano, 1972; Wigstrom, 1973; Willeshaw, Buneman, and Longuet-Higgins, 1969). The simplest of these are based on the properties of correlation matrices, and all of them exhibit interesting and suggestive forms of content addressability, generalization, and error tolerance. There have also been numerous discussions of the possibility that these forms of memory structures may

^{*} This section will appear in Biological Cybernetics, 1981
(authors A. G. Barto, R. S. Sutton, and P. S. Brouwer).

provide models of biological memories. In all of these studies, the storage process is one in which a series of "keys" and "patterns" are repeatedly presented to the memory network which stores the key-pattern associations.

As models of memory, these associative memory structures suggest how a rapprochement might be reached between connectionistic, locationalistic views of memory and Gestalt, mass action views (e.g., Freeman, 1975; John and Schwartz, 1978). Associative memories use learning rules that are connectionistic in character yet need not store information in localized form. However, as models of learning they exhibit only a very simple form of open-loop learning. Since the desired response (the pattern to be reproduced) and the stimulus intended to elicit that response (the key) are both explicitly presented to the system during the training phase, these studies do not address the case of learning in which neither the associative memory nor the environment knows the desired response.

In this section we describe an associative memory structure, called an Associative Search Network or ASN, which is not told by some outside process (e.g., a "teacher") what pattern it is to associate with a given key. Instead, for each key, the network must search for that

pattern which maximizes an external payoff or reinforcement signal. The pattern that will produce the maximal payoff for each key is never available to the system. It operates by generating an output pattern, receiving an evaluation from its environment in the form of a scalar level of payoff or reinforcement, updating the contents of its memory, and then repeating this "generate-and-test" procedure. As this kind of learning proceeds, each key causes the retrieval of better choices for the pattern to be associated with that key. What gets stored in the associative memory is a result of reinforcement feedback through the environment. By eliminating the need for a "teacher" to explicitly provide the pattern to be stored, the ASN effectively solves a central problem faced by an adaptive system. No part of the system need have a priori knowledge about what associations are best.

This type of learning should not be confused with what is commonly called "unsupervised learning" or "learning without a teacher." These labels refer to the problem of clustering input patterns according to a given measure of similarity so that members of each cluster are more similar to one another than they are to members of other clusters. Like the learning exhibited by the associative memory structures cited above, this type of learning is open-loop: any consequences of the system's actions are irrelevant.

The type of learning exhibited by the ASN should also not be confused with learning in which an error rather than reinforcement or payoff is returned by the environment. There are several important differences between error-signal and reinforcement learning, but the most important one to be noted here is that for an associative memory system, the error must be a vector giving the signed component-wise error of the system's response. The reinforcement signal returned to the ASN, on the other hand, is a scalar which is just the environment's evaluation of the system's response. The fact that the ASN is able to learn to produce optimal output vectors based on scalar environmental feedback should be kept firmly in mind. This type of learning has been called "Learning with a critic" by Widrow et al. (1973). A critic need not know what each optimal response is in order to provide useful advice.

The ASN combines two types of learning which are usually only considered separately. First, it solves a pattern recognition problem by learning to respond to each key with the appropriate output pattern. This is the problem solved by the associative memory systems described in the literature. The method used is similar to stochastic approximation pattern recognition methods (see, for example, Duda and Hart, 1973, for a good discussion of these techniques). At the same time, the ASN uses a different

type of learning to actually find what output pattern is optimal for each key. It effectively performs a search using a stochastic automaton method to maximize a payoff or reinforcement function. Stochastic automaton search methods originated in the work of Tsetlin (1971) and are reviewed by Narendra and Thathachar (1974). Other systems capable of performing this kind of search do not perform the pattern recognition task. For example, the ALOPEX system of Harth and Tzanakou (1974), to which the ASN is closely related, performs a search but is not sensitive to different input patterns or keys and thus is not an associative memory. The learning the ASN accomplishes solves both the search and the pattern recognition problem in a simple and effective way.

Although learning systems capable of solving both types of problems have been discussed in the adaptive system theory literature (Mendel and McLaren, 1970), these systems do not have the error tolerance and generalization capabilities of distributed associative memories. The only neural theory which contains this synthesis is that of Klopf (1972, 1979, 1981). Klopf emphasizes closed-loop reinforcement learning and correctly points out that, despite common opinion to the contrary, it has been largely neglected by neural theorists. The results presented here demonstrate the significance and novelty of Klopf's theory. We will discuss the ASN in light of Klopf's theory in some

detail below. Also closely related is the notion of "bootstrap adaptation" of Widrow et al. (1973).

5.2 The Associative Search Problem

Figure 5.1 shows an ASN interacting with an environment E. At each time t , E provides the ASN with a vector $X(t) = (x_1(t), \dots, x_n(t))$, where each $x_i(t)$ is a positive real number, together with a real valued payoff or reinforcement signal $z(t)$. The ASN produces an output pattern $Y(t) = (y_1(t), \dots, y_m(t))$, where each $y_i(t) \in \{0, 1\}$, which is received by E. The problem the ASN is designed to solve can be stated informally as follows. Each vector $X(t)$ provides information to the ASN about the condition or state of its environment at time t , or, viewed in another way, provides information about the sensory context or situation in which the ASN should act. We call each $X(t)$ a context or situation vector. Different actions, or output patterns, are appropriate in different contexts. As a consequence of performing an action in a particular context, the ASN receives from its environment, in the form of a payoff or reinforcement signal, an evaluation of the appropriateness of that action in that context. The ASN's task is to act in each context so as to maximize this payoff. We are using the term context merely to refer to the environmental background in which an action is taken. We do not wish to

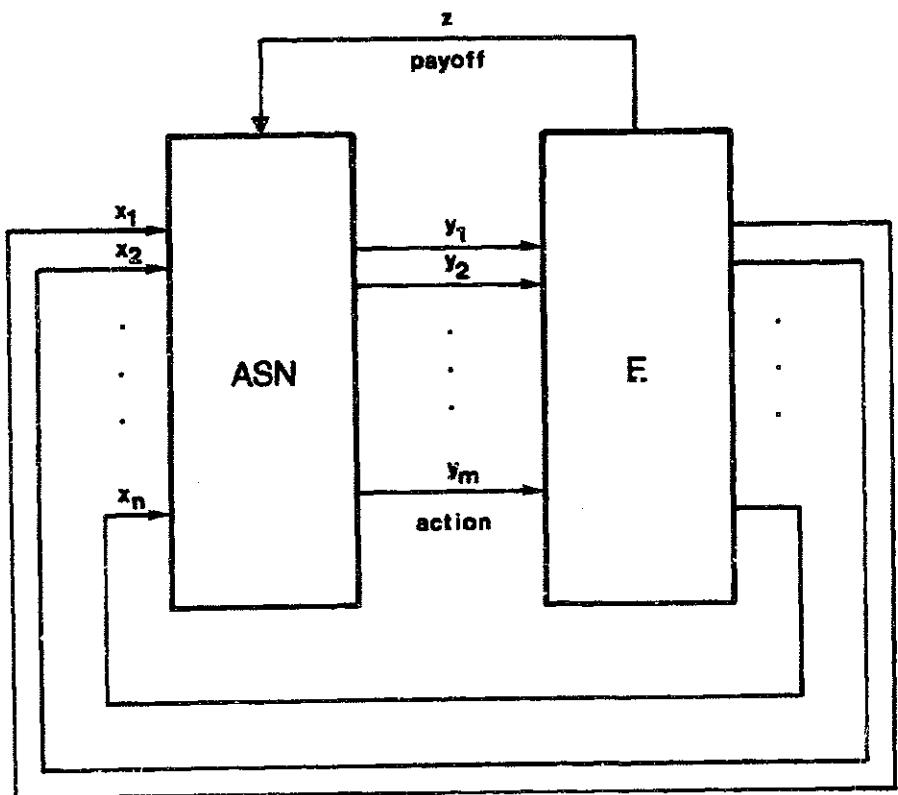


FIGURE 5.1. An ASN interacting with an environment E. The ASN receives context signals x_1, \dots, x_n and a payoff or reinforcement signal z from E and transmits actions to E via output signals y_1, \dots, y_m .

imply that all of this term's more specialized meanings are applicable here.

More formally, we assume that $X(t)$ belongs to a finite set $X = \{X_1, \dots, X_k\}$ of context vectors and that to each $X_a \in X$ there corresponds a payoff or reinforcement function Z_a . Assuming that E always evaluates an output vector in one time step, if $X(t) = X_a$, then $z(t+1) = Z_a(Y(t))$. We say that E provides a training sequence over X if it implements an infinite sequence of payoff functions and emits the corresponding sequence of context vectors

$$X_{i_1}, X_{i_2}, \dots, X_{i_g}, \dots$$

such that each $X_{i_g} \in X$ and each element of X occurs infinitely often (Nilsson, 1965). The associative search problem is solved if, after some finite portion of a training sequence, the ASN responds to each $X_a \in X$ with the output pattern $Y_a = (y_1^a, \dots, y_m^a)$ which maximizes Z_a . Generalizations of this problem are discussed below.

5.3 The Basic Adaptive Element

An ASN consists of a number of identical adaptive elements each determining a component of the system's actions. It is useful to describe first a single element which can be regarded as the simplest ASN ($m=1$). Figure 5.2

shows an adaptive element interacting with an environment E. The element has n context input pathways x_i , $i = 1, \dots, n$, one payoff or reinforcement pathway z, and one output y. Associated with each context pathway x_i is a real valued weight w_i with value $w_i(t)$ at time t. Let $W(t)$ denote the weight vector at time t. Let $s(t)$ denote the weighted sum at time t of the context inputs. That is,

$$s(t) = \sum_{i=1}^n w_i(t)x_i(t) = W(t) \cdot X(t).$$

The output $y(t)$ is determined from $s(t)$ as follows:

$$y(t) = \begin{cases} 1 & \text{if } s(t) + \text{NOISE}(t) > 0 \\ 0 & \text{otherwise,} \end{cases} \quad (5.1)$$

where NOISE is a random variable with mean zero normal distribution. The sum s therefore biases the element's output (cf. Harth and Tzanakou, 1974): Positive s making it more likely to be 1, and negative s making it more likely to be 0.

The weights w_i , $i = 1, \dots, n$, change according to a discrete time iterative process. At each time step, each weight is updated according to the following equation: for $i = 1, \dots, n$,

$$w_i(t+1) = w_i(t) + c[z(t) - z(t-1)][y(t-1) - y(t-2)]x_i(t-1) \quad (5.2)$$

where c is a constant determining the rate of learning.

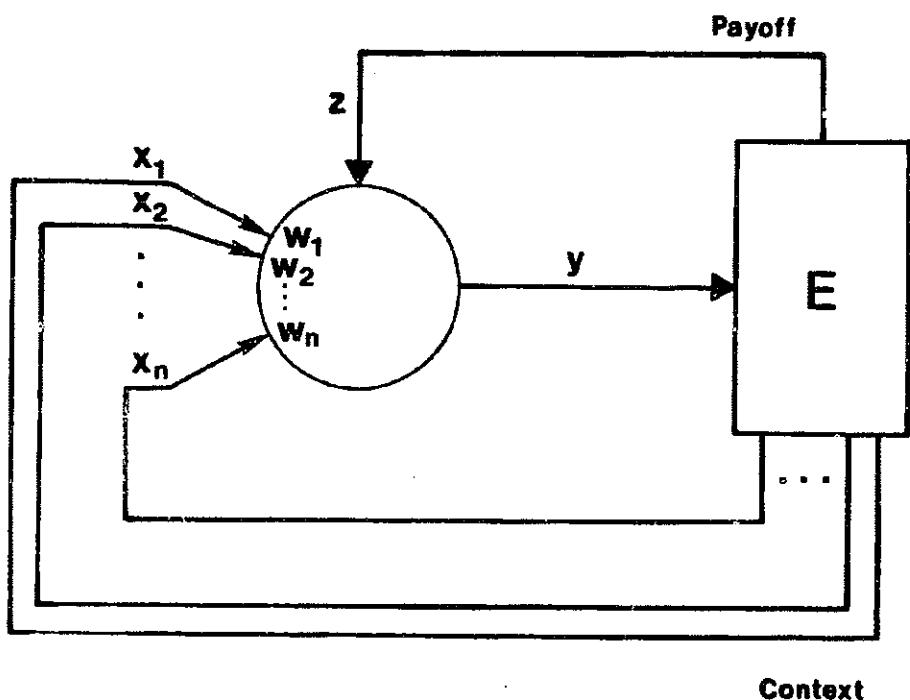


FIGURE 5.2. The simplest ASN: A single adaptive element interacting with an environment E.

Other rules also work, but this is one of the simplest. Also for simplicity the response latency for the element is zero; that is, there is no delay between input and output. This causes no difficulties here because we do not consider recurrent connections within a network. In other variants, the inputs need not be positive, the output signal y need not be binary, and the noise need not be normally distributed. If the rightmost term $x_i(t-1)$ were removed from Equation 5.2, the resulting learning rule would be essentially that used by Harth and Tzanakou (1974) in the ALOPEX system.

To understand how Equation 5.2 works, consider a simple example. Suppose a positive context signal was present on pathway x_i at some time $t-1$, signaling some condition of the environment. Suppose also that $y(t-1) = 1$ while $y(t-2) = 0$ (that is, the element "turned on" at time $t-1$), perhaps due to an excitatory effect of signal x_i or perhaps by chance. Then, if the payoff signal z increases from time $t-1$ to t (possibly as a result of the element's action), w_i will increase. Since $w_i(t)x_i(t)$ is used to compute $y(t)$, the increased weight w_i will make it more likely (other things being equal) that y will be 1 when signal x occurs in the future. Similarly, if z decreases following the element's action, w_i will decrease thereby decreasing the probability that y will be 1 when signal x_i occurs again. Consequently,

if turning on in a specific context is followed by an increase in payoff, the element will be more likely to turn on (or stay on) in that context in the future. Other cases can be analysed similarly: If going off in a context leads to a payoff increase, then the probability of being off in that context increases. Of course, a pathway can participate in signaling a large number of different contexts. This is where the associative memory properties become relevant.

For an ASN consisting of a single adaptive element, the search for the optimal action for each context vector is not very difficult since the ASN has only two actions. However, a property of the adaptive element that is essential for its use as a component in a larger ASN is that it is capable of operating effectively in environments with random payoff response characteristics. If for each context the output of the adaptive element only determines a probability for the payoff value, the adaptive element is capable of acting so as to increase its expected payoff value. It is beyond the scope of the present discussion to thoroughly discuss these aspects of the adaptive element's behavior. The relevant theory is that of stochastic automaton learning algorithms, and the reader is referred to the review by Narendra and Thathachar (1974).

5.4 The Problem of Context Transitions

According to Equation 5.2, the adaptive element uses the change in the payoff signal z as a factor determining weight changes. However, when the context changes, that is, when the payoff function implemented by E changes, the change in the value of z is due to the change in payoff function as well as the adaptive element's action. The difficulty this creates can be clearly appreciated by considering the worst case in which the payoff function changes at every time step. Consecutive values of z in this case result from evaluating different functions rather than the same function twice, and hence they do not provide useful gradient information about any single payoff function. Unless the payoff functions implemented by E vary smoothly over time, one would not expect an adaptive element operating according to Equations 5.1 and 5.2 to be capable of solving an associative search problem.

Two methods of solving the problem of context transitions are used in the examples which follow. One is to require E to implement each payoff function, and emit the corresponding context vector, for at least two consecutive time steps and, when transitions do occur, to set the learning constant c to zero so that the change in payoff due to the transition has no effect. This procedure requires

either a priori knowledge about when transitions occur or a mechanism for detecting transitions. Such mechanisms can be devised (Didday, 1976, and Grossberg, 1976, discuss this problem and propose neurally plausible methods). For simplicity in some of the examples to follow, we set c to zero "manually" when a transition occurs.

In other examples, however, we use a method that does not require transitions to be known or detected. Suppose the adaptive element produced action $y(t-1)$ in response to context vector $X(t-1)$. Instead of comparing the resulting payoff $z(t)$ with $z(t-1)$, which may have been determined by a different payoff function, we compare it with the payoff "expected" for acting in context $X(t-1)$. If a higher than expected value is obtained, then the action which produced it is made more likely to occur in that context again. In this way, the gradient of each payoff function can be estimated from samples which do not occur consecutively in time. Instead of computing weight values according to Equation 5.2, we use the following rule:

$$w_i(t+1) = w_i(t) + c[z(t) - p(t-1)][y(t-1) - y(t-2)]x_i(t-1) \quad (5.3)$$

which differs from Equation 5.2 by the substitution for $z(t-1)$ the value $p(t-1)$ predicted for $z(t)$ given $X(t-1)$.

We use another type of adaptive element to compute

$p(t-1)$ from $X(t-1)$. This element is a variant of one described in Section 4 and proposed as a model of classical conditioning. It learns to anticipate the payoff rather than to maximize it, and we call it a predictor. The predictor has n context pathways x_i , $i = 1, \dots, n$, one payoff pathway z , and one output pathway p . Associated with each context pathway x_i is a variable weight w_{pi} . The output at time t is

$$p(t) = \sum_{i=1}^n w_{pi}(t)x_i(t).$$

The weights change over time according to the following equation: For $i = 1, \dots, n$,

$$w_{pi}(t+1) = w_{pi}(t) + cp[z(t) - p(t-1)]x_i(t-1)$$

where cp is a learning constant determining the rate of learning. This rule is identical to Equation 5.3 but with $y(t-1)-y(t-2)$ fixed at the value 1. This element implements a stochastic approximation method for finding weights (if such weights exist) such that $p(t-1) = z(t)$ for all t . In other words, the predictor output anticipates by one time step the payoff supplied by the environment. If a linear prediction is not possible, Equation 5.4 will find the best-least-square linear prediction if cp is allowed to decrease over time. See Duda and Hart (1973) and Kasyap, Blaydon, and Fu (1970) for good discussions of these methods.

5.5 A Network

Figure 5.3 shows an ASN consisting of m adaptive elements and one predictor. Each context pathway from the environment connects to each adaptive element and to the predictor, as does the payoff pathway z . The adaptive element weights form an $m \times n$ matrix $W = (w_{ij})$ where w_{ij} is the weight of the i -th adaptive element for the j -th context pathway. The random variables NOISE for each element are independent and identically distributed, and the learning constants are the same for each element.

While the training sequence is being presented, each adaptive element comprising the ASN faces the problem discussed above of maximizing each payoff function. Due to the dependence of each element's payoff on the joint activity of all the elements' activity, each element's payoff appears to have a random component since it depends on the unknown outputs of the other adaptive elements comprising the ASN. As a result of the capability of each adaptive element to increase its expected payoff when interacting with an environment having random response characteristics, an ASN consisting of any number of adaptive elements can solve the corresponding associative search problem under certain conditions.

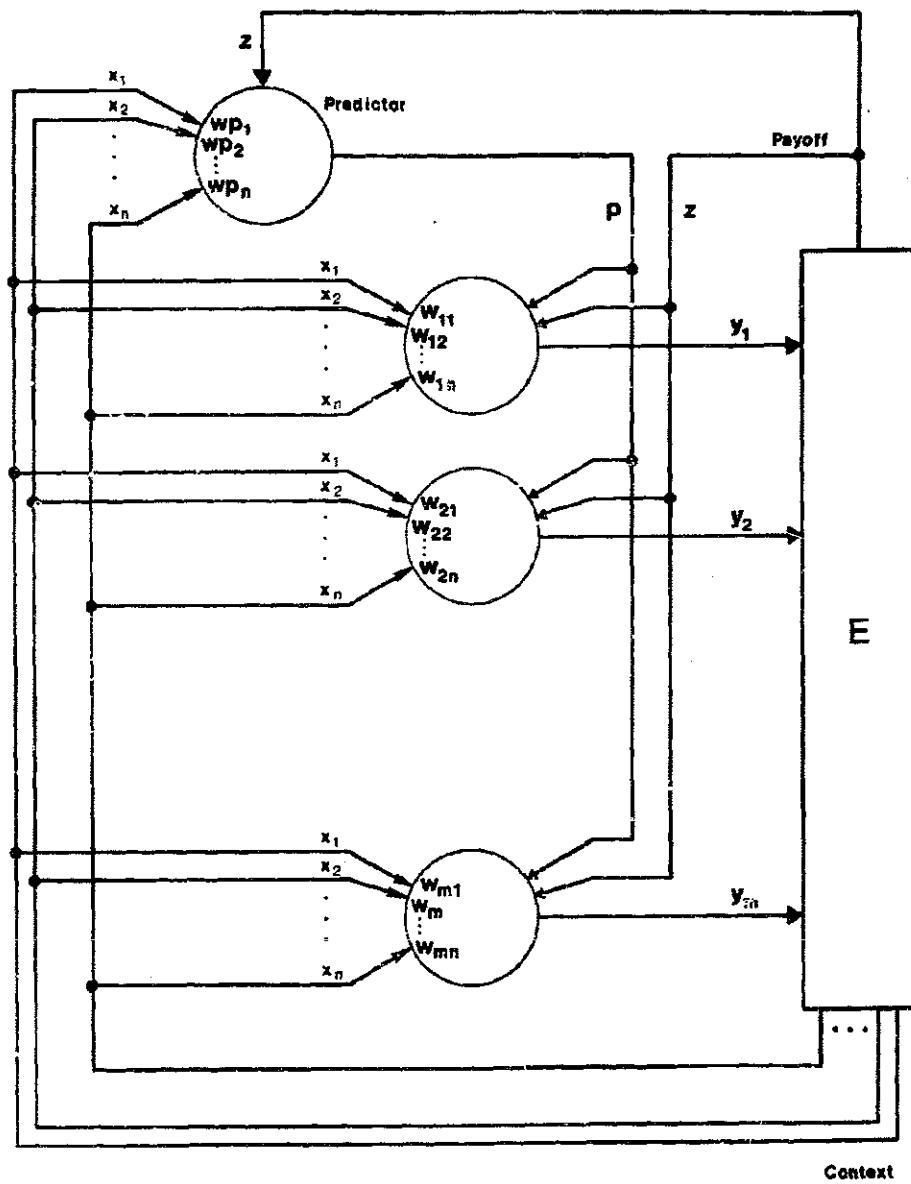


FIGURE 5.3. An ASN consisting of m adaptive elements and one predictor. The adaptive element weights form an $m \times n$ associative matrix.

For each context vector, the associative search problem is an example of what is known in the theory of learning automata as a cooperative game of learning automata (Narendra and Thathachar, 1974). Unlike other learning automata studied, however, the ASN solves such a problem for each context vector. By combining notions from the theory of cooperative games of learning automata and the theory of pattern recognition, we can formulate a conjecture about the conditions under which the ASN as described here can solve the associative search problem. For each i , $i = 1, \dots, m$, let

$$x_i^0 = \{x_a \in X | y_i^a = 0\}$$

$$x_i^1 = \{x_a \in X | y_i^a = 1\}.$$

That is, x_i^0 (x_i^1) is the set of all context vectors in which it is optimal for element i to produce output 0 (1). The sets x_i^0 and x_i^1 are linearly separable if there exists a real vector $w_i = (w_{i1}, \dots, w_{in})$ such that

$$w_i \cdot x < 0 \text{ if } x \in x_i^0$$

$$w_i \cdot x > 0 \text{ if } x \in x_i^1.$$

We conjecture that for any $n, m > 0$, there exist ASN parameters (c, c_p , and the variance of the random variables) such that it can solve the associative search problem with as high a probability as desired if 1) each Z_a is unimodal (i.e., does not possess suboptimal "peaks") and 2)

x_i^0 and x_i^1 are linearly separable for each $i = 1, \dots, m$. The performance of learning automata in optimizing multimodal functions is a topic of current research.

Once this task is solved, the ASN functions as an associative memory similar to those discussed in the literature. For example, if a degraded context vector is presented, then the ASN can still perform an appropriate action if the degraded context vector is still sufficiently distinctive. Similarly, the ASN will produce actions in situations never before encountered by acting in a way appropriate in similar situations which it has experienced in the past. The ASN also exhibits the same resistance to damage shown by distributed associative memories (see Wood, 1978). In addition it is possible to prime the associative matrix with information likely to be useful for specific problem domains.

We note that if our conjecture is correct, perfect ASN performance does not require orthogonal context vectors. Associative memories have been discussed by Amari (1977a, b) and Kohonen and Oja (1976) which are able to exhibit perfect recall if the keys are linearly independent but not orthogonal. Amari (1977a, b) calls this orthogonal learning since it requires the orthogonalization of the set of keys. It can be shown that if the context vectors X_1, \dots, X_k are

linearly independent, then x_i^0 and x_i^1 are linearly separable for each $i = 1, \dots, m$. This implies that if our conjecture is true, the ASN can solve the associative search problem if each Z_a is unimodal and the context vectors are linearly independent. This is an instance of orthogonal learning, but, as discussed above, it differs in that the ASN does not require the desired response for each key to be explicitly provided.

5.6 Examples

For illustrative purposes we let each payoff function Z_a in the following examples be a simple linear function of the ASN actions. To each context vector X_a is associated a vector $Y_a = (y_1^a, \dots, y_m^a)$ where $y_i^a \in \{-1, 1\}$. We define Z_a as

$$Z_a(Y) = Y \cdot Y_a$$

so that Z_a is maximized when each adaptive element i , $i = 1, \dots, m$, is "on" if $y_i^a = +1$ or "off" if $y_i^a = -1$. That is, Z_a is maximized by $Y = (Y_a + 1)/2$. We use the symbol Y_a to denote both the $1, -1$ valued vector Y_a and the binary vector $(Y_a + 1)/2$ since no confusion is likely to arise. Computing Z_a in this manner implies that if an adaptive element "turns on" in a context in which it should be on, or if it "turns off" in a context in which it should be off, then the value of Z_a increases by 1 (assuming the other elements don't

change their actions). Similarly, "turning on" when off is best, or "turning off" when on is best, decreases Z_2 by 1. We do not claim that the optimization of such a simple linear function is a difficult task. Our intent here is to illustrate that a search is in fact performed by the ASN. More research is required to delineate the search capabilities of the ASN and related structures. In each of the following examples, the adaptive element learning constant $c = .03$ and the standard deviation of each random variable is .1. In the cases using the predictor, $cp = .1$.

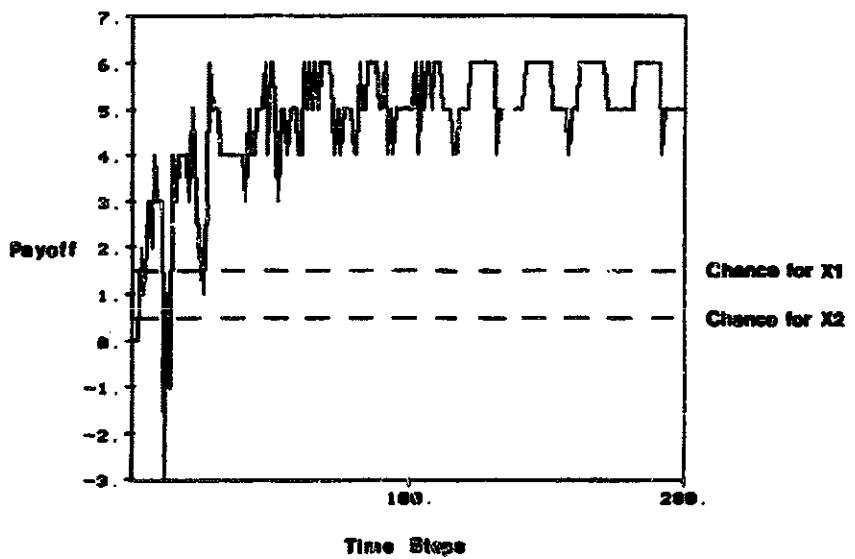
Example 1

Figure 5.4 shows ASN behavior for the simplest case of two orthogonal context vectors X_1 and X_2 with $n = 8$ and $m = 9$. The optimal output patterns are determined by Y_1 and Y_2 (Figure 5.4a). Notice that $Z_1(Y_1) = 6$ and $Z_2(Y_2) = 5$ so that a higher payoff is obtainable in context 1. The contexts were alternately presented, each held constant for 10 time steps. A predictor was not used. In order to prevent the transition from one context to another from providing misleading information, the learning constant c was momentarily set to zero while the contexts changed.

The dashed lines in Figure 5.4b show the payoffs which could be expected in each context for output patterns generated purely by chance. The payoff actually received by

$$\begin{array}{l}
 X_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad Y_1 = \begin{bmatrix} -1 \\ 1 \\ 1 \\ 1 \\ -1 \\ 1 \\ 1 \\ 1 \\ -1 \end{bmatrix} \\
 X_2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad Y_2 = \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \\ 1 \\ -1 \\ -1 \\ 1 \\ 1 \end{bmatrix}
 \end{array}$$

(a)



(b)

FIGURE 5.4. Example 1. a) Two orthogonal context vectors X_1 and X_2 and the corresponding optimal output patterns Y_1 and Y_2 . b) Graph of payoff received by the ASN during a training sequence in which contexts were presented alternately, each held constant for 10 time steps.

the ASN increases over time and attains the optimal values for each context; i.e., 6 for context X1, 5 for context X2. After learning, the presentation of a context vector immediately "keys out" the pattern optimal for that context. Unlike other associative memory systems, however, the optimal patterns were never directly available to the system. Since the context patterns in this case have totally disjoint regions of nonzero values, the more interesting associative aspects of the system are not demonstrated. The resultant associative matrix simply stores the separate associations.

Figure 5.5 shows the behavior of the ASN for exactly the same problem as illustrated in Figure 5.4 with the exception that the learning constant c was not set to zero for context transitions. Learning occurs, but the almost perfect behavior shown in Figure 5.4b is not attained even after 500 time steps. The reason for this is that the transition from X1 to X2 tends to penalize elements which may have been correctly responding to X1 since the payoff tends to decrease at the transition.

Figure 5.6 illustrates the behavior of the ASN with a predictor for the same problem shown in Figures 5.4 and 5.5. The learning curve (Figure 5.6a) is comparable to that obtained with c set to zero during transitions.

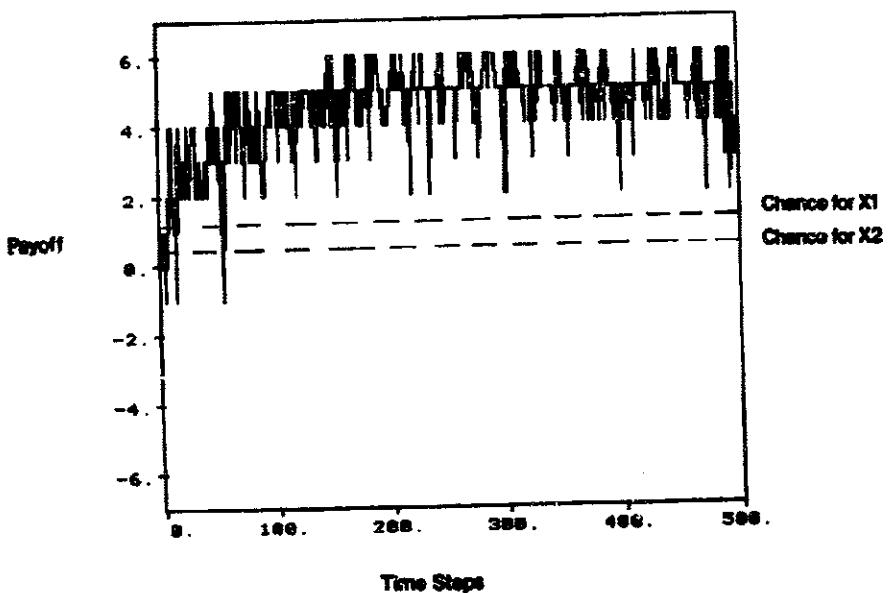
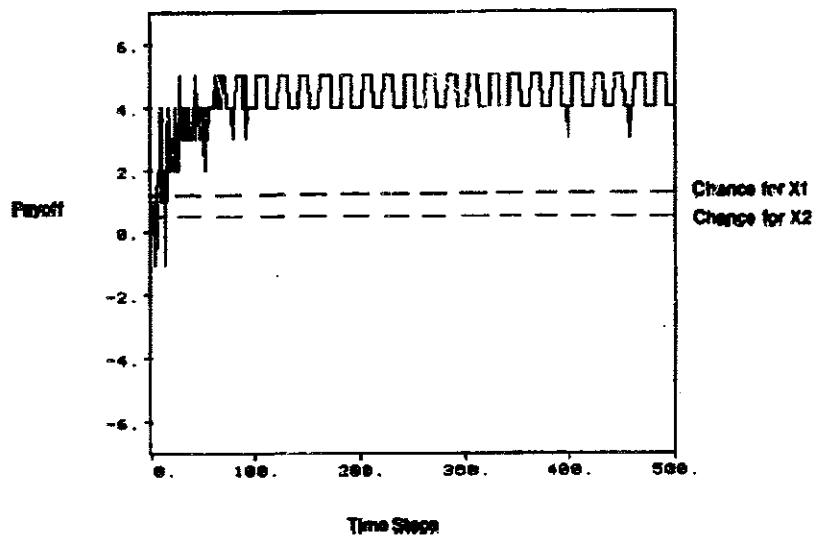
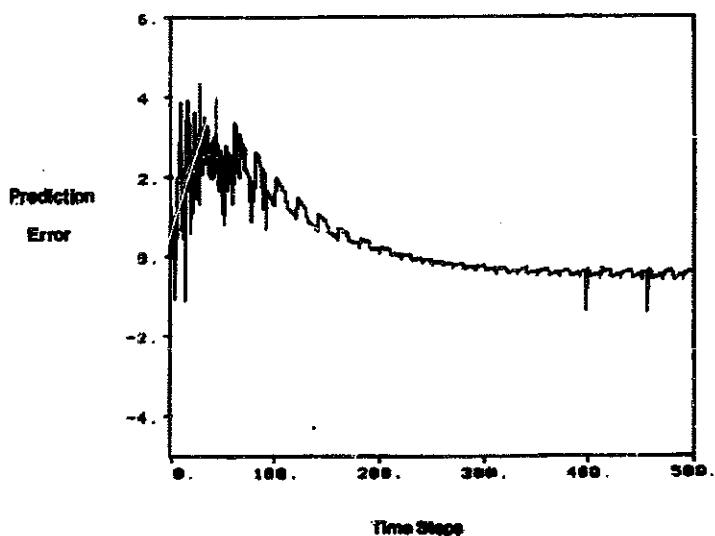


FIGURE 5.5. The ASN payoff for the training sequence illustrated in Figure 5.4 but with the learning constant held nonzero throughout. The perfect behavior shown in Figure 5.4a is not attained.



(a)



(b)

FIGURE 5.6. a) The ASN payoff for the training sequence illustrated in Figure 5.4 but with the use of a predictor.
b) Prediction error $p(t) - z(t+1)$.

Figure 5.6b shows the predictor error $p(t) - z(t+1)$ during the training sequence. The predictor comes to successfully predict that the highest payoffs in contexts X_1 and X_2 are respectively 6 and 5. Transitions from X_1 to X_2 do not penalize elements correctly responding to X_1 since the payoff drop is "expected". Notice in Figure 5.6 the errors committed approximately at time steps 400 and 450. Since we use normally distributed random variables to drive the search, there always remains a nonzero probability that an element will perform either action.

Example 2

Here $n = 3$, $m = 25$, and four non-orthogonal but linearly independent context vectors are considered (Figure 5.7a). The optimal output patterns Y_1, \dots, Y_4 are shown as 5×5 arrays, but should be thought of as "actions" and not as visual images. Again, each context was presented for 10 consecutive time steps, with the sequence repeating. No predictor was used. The learning constant was set to zero during context transition. After sufficient learning each context vector causes the retrieval of the optimal output pattern. This occurs even though the context vectors do not form an orthogonal set. Figure 5.7b shows the learning curve for context X_1 . The abscissa gives cumulative times steps in which context X_1 was present. An ASN using a predictor has essentially the same behavior.

Example 3

With the associative matrix W containing the values obtained after training in Example 2, context vector X_1 was corrupted by additive noise and presented to the ASN (Figure 5.8a). As for other associative memories, keys corrupted by noise cause retrieval of patterns similar to the desired ones provided the corrupted key remains sufficiently distinguishable from the others. The pattern retrieved using the corrupted version of X_1 resembles the stored pattern Y_1 . For the ASN, however, the retrieved pattern is just the initial guess (Figure 5.8a) for the optimal pattern and the search resumes. Like most search procedures, the time to convergence for the ASN is reduced if the initial guess is close to the optimal pattern. Hence, with the corrupted X_1 being presented to the ASN and Y_1 still the best output pattern, the ASN quickly corrects its response (Figure 5.8b). At the conclusion of the search, the corrupted version of X_1 is able to cause the immediate retrieval of Y_1 .

Example 4

Again with the associative matrix containing the values obtained by training in the four contexts of Example 2, a fragment of X_1 is presented as a context vector (Figure 5.9a). The pattern retrieved again acts as an initial guess

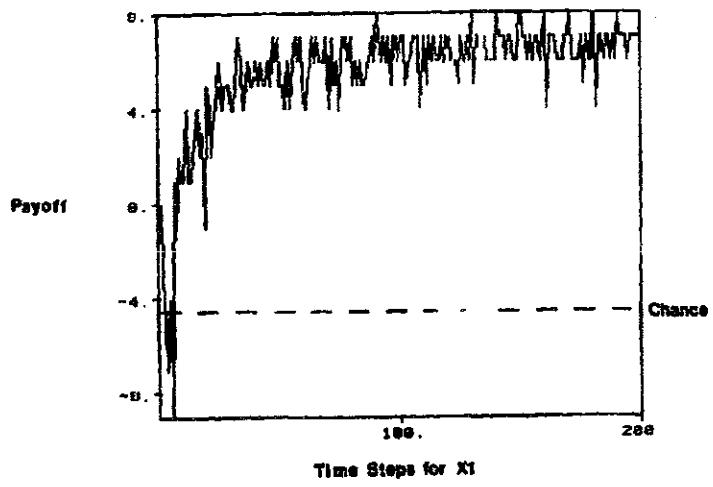
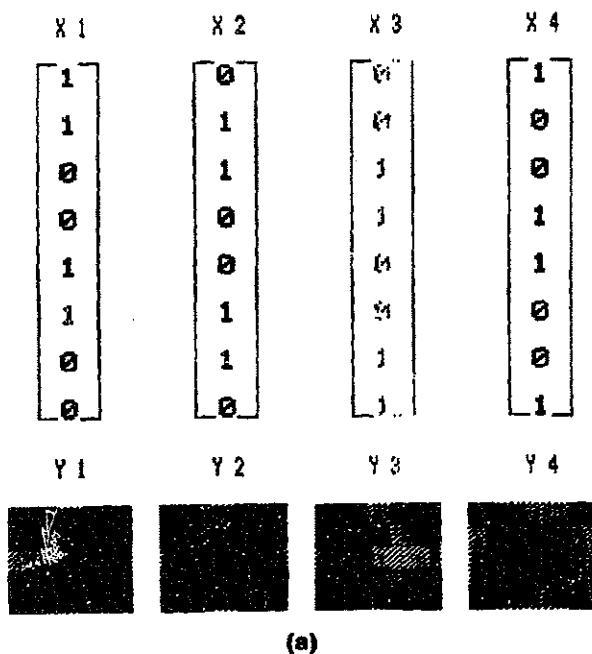
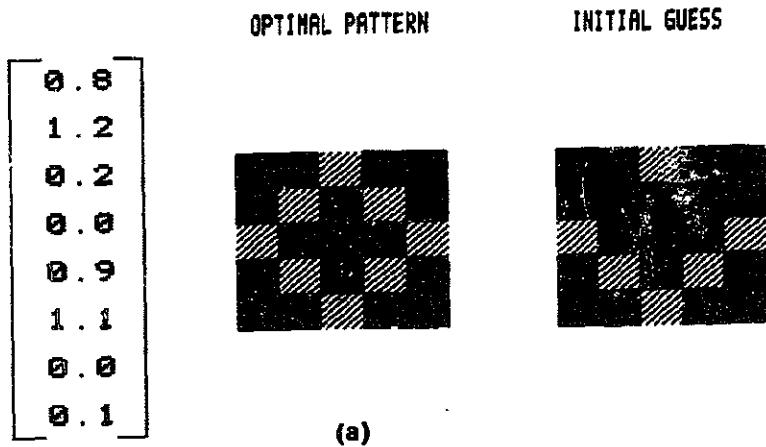
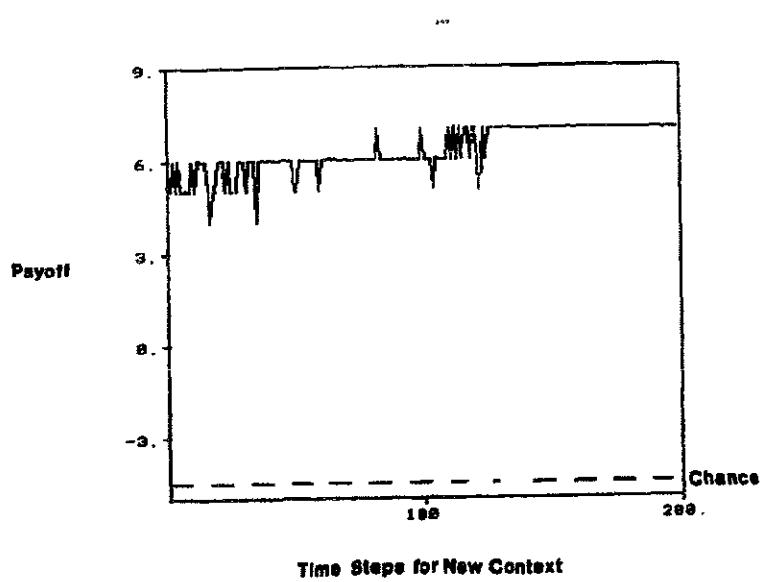


FIGURE 5.7. Example 2. a) Four non-orthogonal but linearly independent context vectors and their corresponding optimal output patterns. b) ASN payoff for time steps in which context vector X₁ is present. There is a similar curve for each context vector.

CONTEXT



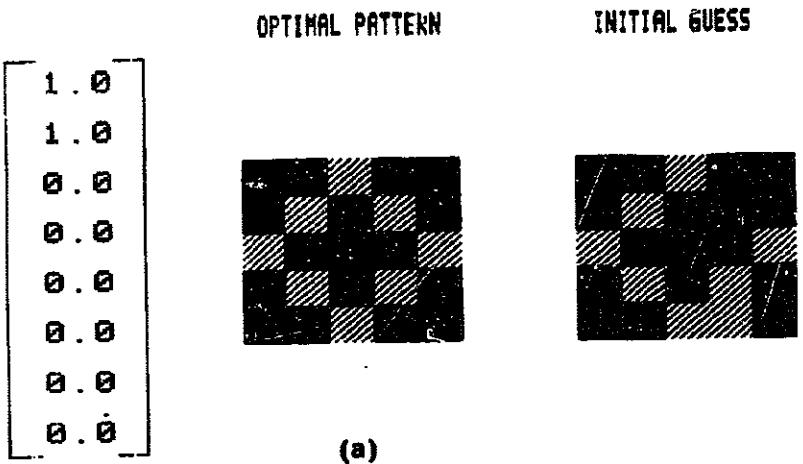
(a)



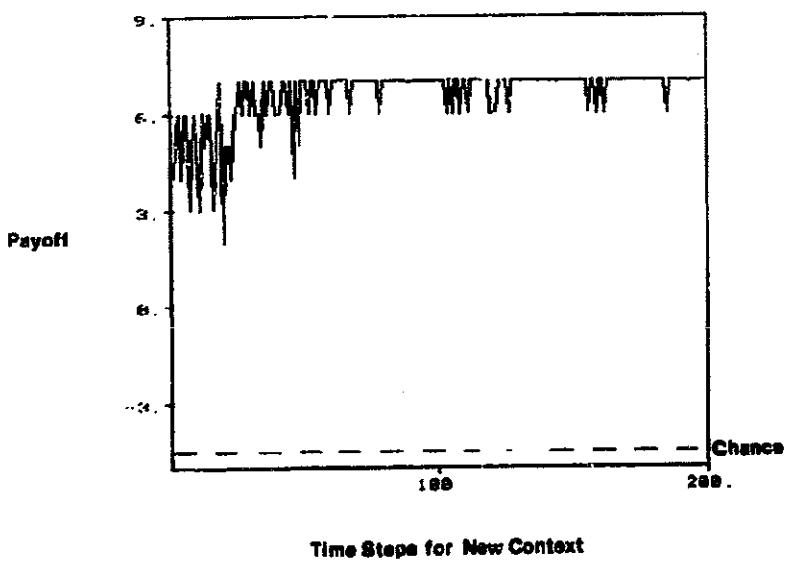
(b)

FIGURE 5.8. Example 3. a) The corrupted context vector, the optimal output pattern, and the ASN's initial guess.
b) ASN payoff as it searches for the optimal output pattern.

CONTEXT

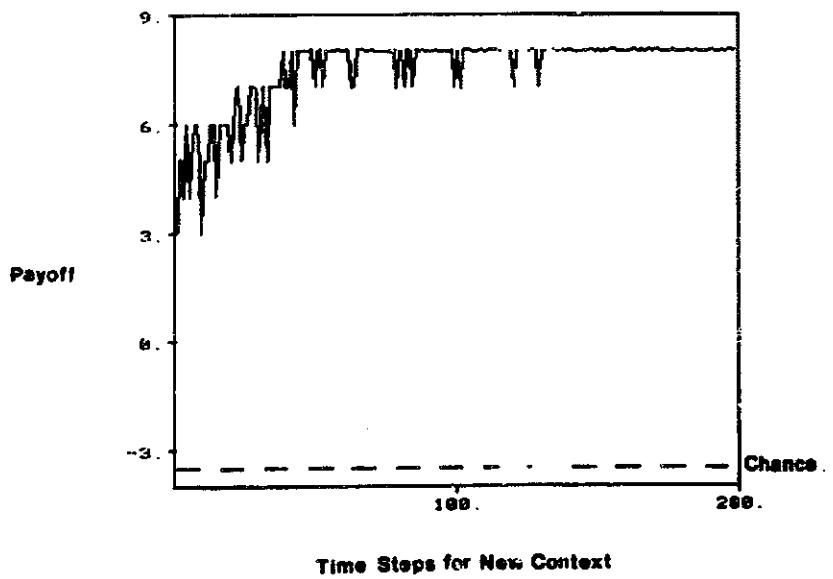
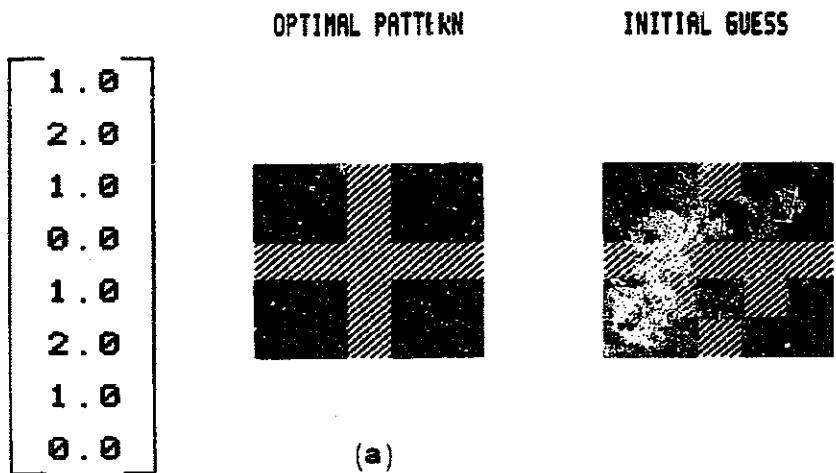


(a)



(b)

FIGURE 5.9. Example 4. a) The fragment of X_1 , the optimal pattern, and the initial guess. b) ASN payoff as the search continues.



(b)

FIGURE 5.10. Example 5. a) The context vector $X_1 + X_2$, the optimal output pattern Y_2 , and the ASN's initial guess.
b) ASN payoff as the search continues.

and the ASN corrects it under control of environmental feedback (Figure 5.9b).

Example 5

Here the sum of the two context signals X_1 and X_2 of Figure 5.7a is presented as a context vector to the ASN, but the payoff function is the one previously signalled by X_2 (that is, Y_2 is best). In this case, the initial guess is a combination of the patterns Y_1 and Y_2 (Figure 5.10a). Again the search process brings the initial guess to the optimal pattern (Figure 5.10b).

5.7 Neural Search

The ASN arose from our investigation of the neural hypothesis of Klopff (1972, 1979, 1981). He hypothesized that neurons try to maximize their level of membrane depolarization by changing synaptic effectiveness in the following way: Whenever a neuron fires, those synapses that were active during the summation of potentials leading to the discharge become eligible to undergo changes in their transmission effectiveness. If the discharge is followed by further depolarization, then the eligible excitatory synapses become more excitatory. If the discharge is followed by hyperpolarization, then eligible inhibitory

synapses become more inhibitory. In this way a neuron will become more likely to fire in a situation in which firing is followed by further depolarization and less likely to fire in a situation in which firing leads to hyperpolarization.

The basic adaptive element operating according to Equation 5.2 is very similar to Klopf's model of a neuron. The term $x(t-1)$ in Equation 5.2 corresponds to Klopf's eligibility. A weight can change at time t only if there was activity on its pathway at $t-1$; that is, $x(t-1) = 0$. More general forms of eligibility can be implemented by replacing this term with a more prolonged trace of activity as is discussed in Section 4. The restricted form of eligibility used here is suitable because E always evaluates an output pattern in a single time step. The idea of eligibility is essential for the search behavior of an adaptive element since it permits the consequences of actions to influence the probability of these actions in the future. This cannot be accomplished by a Hebbian-type rule which associates simultaneous, or nearly simultaneous, signals with no sensitivity to which occurred earliest.

Unlike Klopf's hypothesized neuron, the adaptive element presented here tends to maximize a specialized payoff or reinforcement signal (z) rather than what would correspond to membrane potential (s). There are several

interesting consequences of a rule that tends to maximize s . It permits secondary reinforcement to occur whereby the occurrence of a previously rewarded context itself is rewarding, and it may permit a single adaptive element to perform both the search and prediction tasks, eliminating the need for a separate predictor element. In this section we have focused only on the simpler case in which there is a specialized payoff or reinforcement signal.

The adaptive element presented here is an illustrative example of a class of adaptive mechanisms, some of which are more closely related to Klopf's hypothesis, and should not be literally interpreted as a model of a single neuron. In fact, we have purposefully referred to it as an adaptive element rather than a neural model. We do wish to suggest, however, that the general form of stochastic, closed-loop, optimization learning realized by the adaptive element merits close experimental investigation. Theory has shown that stochastic search procedures can be very effective means for the optimization of functions about which little is known. This capability, combined with pattern recognition capabilities, leads to considerable adaptive power. As a neural hypothesis, the adaptive element suggests that the stochastic component of neural discharge might perform the function of stochastic search. A closely related adaptive element is discussed with respect to

behavioral and neurophysiological data in Section 4.

5.8 Sensorimotor Control Surfaces

It has been suggested that associative memories might provide effective means for the storage of sensorimotor associations required for sensory guided motor behavior (Albus, 1979). However, in every case there is the requirement for a signal to be present giving the "desired response" in order to form the correct sensorimotor association. Yet this kind of information is usually not available to an organism nor easy to obtain. After considerable experience in a given set of sensory contexts, the "desired response" for each context might become known through a learning process. But the associative memory structures proposed in the literature are not able to perform this type of learning. Their structure suggests how associations might be stored but does not address the very important questions concerning what information is chosen for storage. The ASN suggests how such questions might be explored.

Sensorimotor tasks provide natural examples of the type of problem the ASN is capable of solving. Sensory context is provided by exteroceptive and interoceptive stimulus patterns, and output patterns provide control signals to

motor systems. Global reinforcement systems might provide information analogous to the ASN payoff signal. The associative matrix formed would implement a sensorimotor control surface. This interpretation of the ASN task suggests that research should continue in order to extend the ASN's capabilities in several different ways. 1) Most complex control tasks require nonlinear control surfaces. Elaboration of the ASN to permit the formation of nonlinear associations can be accomplished in the same manner as suggested for other associative memories in the literature (Poggio, 1975). 2) Most sensorimotor tasks have the property that the context which occurs next is partially a function of the control system's action. In the problem discussed in this section, the ASN has no control over which context occurs. An interesting generalization of the ASN task is to require the ASN to control not only the payoff signal but also the context vectors in order to reach a context in which the highest payoff is available. This is a more general learning control problem. 3) The ASN task presented here is simplified by the occurrence of a payoff signal at every time step. In actual sensorimotor learning tasks the reinforcing events occur only occasionally. Secondary reinforcement capabilities would provide a first step toward the solution of this substantially more difficult problem.

5.9 Conclusion

The distributed memory properties of associative memory systems make them particularly interesting learning systems from both biological and theoretical perspectives. Although all associative memory systems described in the literature require the desired response for each key to be provided by some other source, the interesting properties of associative memory systems are not restricted to this form of learning. A more difficult type of learning, which can occur even if no part of the system or of the environment knows the desired behavior, is reinforcement learning. In this form of learning, the environment provides only a performance measure of responses rather than desired responses, making the problem both more difficult for the learning system and less demanding for the environment. The ASN is an associative memory system capable of solving reinforcement learning tasks. Our results illustrate that the important properties of associative memories can be retained by a system capable of this more general and more difficult form of learning.

CHAPTER 5

LANDMARK LEARNING: AN ILLUSTRATION OF ASSOCIATIVE SEARCH *

6.1 Introduction

In Section 5 we defined the associative search problem and presented a system, called an Associative Search Network (ASN), capable of solving it under certain conditions. An ASN incorporates learning rules that have been carefully designed following Klopf's hypothesis that neurons are goal-seeking systems (Klopf, 1972, 1979, 1981). Here we present a simple spatial learning problem as an example of the associative search task. This interpretation illustrates the task in an intuitively clear form, shows how naturally it can arise, and allows the capabilities of a simple ASN to be clearly described. It was not our intention either to model animal spatial learning behavior or to fully exploit the capabilities of an ASN; rather, we wanted to illustrate its capabilities in as simple a problem as we could construct.

* This section will appear in Biological Cybernetics, 1981.

6.2 Associative Search

Figure 5.1 shows an ASN interacting with an environment E. At each time t , E provides the ASN with a vector $X(t) = (x_1(t), \dots, x_n(t))$, where each $x_i(t)$ is a positive real number, together with a real valued payoff or reinforcement signal $z(t)$. The ASN produces an output pattern $Y(t) = (y_1(t), \dots, y_m(t))$, where each $y_j(t) \in \{0,1\}$. The ASN's action Y is received by E. Each vector $X(t)$ provides information to the ASN about the sensory situation at time t in which it acts. After performing an action; that is, after producing an output pattern, the ASN receives (1 time step later) an evaluation from E of the appropriateness of that action for the situation in which it was made. This evaluation is received by the ASN as the value of a payoff or reinforcement signal z . The evaluation alone is not sufficient to determine whether the preceding action was the best possible in the given context. The associative search task is to learn, for each input vector, to perform the action which maximizes the payoff value. In other words, it must learn to perform the best action in each sensory situation. Different actions can be optimal in different sensory contexts. This class of problems is more completely described in Section 5 where it is distinguished from the simpler pattern recognition tasks that can be solved by perceptron-like learning rules.

6.3 Spatial Learning as Associative Search

If an ASN is viewed as controlling the locomotory behavior of an organism in a spatial environment, then input vectors are associated with places in space, and ASN output patterns control movement. We have created a simple spatial environment in which to illustrate this interpretation of the associative search problem and a simple ASN's behavior. Figure 6.1 shows a spatial environment consisting of a central landmark (shown as a tree) surrounded by four other landmarks (shown as disks). Thinking of this as an olfactory environment for a simple organism, we let each landmark possess a distinctive "odor" which can be sensed at a distance. Accordingly, to each landmark is associated a spatial distribution, linearly decreasing with distance from the landmark, which extends as far as the large ellipses shown in Figure 6.1. The asterisk shows the location of the ASN.

When the ASN is in a particular location, its input pattern is determined by its distance from each of the landmarks. We let the central landmark act as an attractant for the ASN by letting its "odor" be the value of the payoff or reinforcement signal z . The other landmarks are "neutral" in that proximity to them is not rewarding to the ASN. An input vector therefore consists of five values

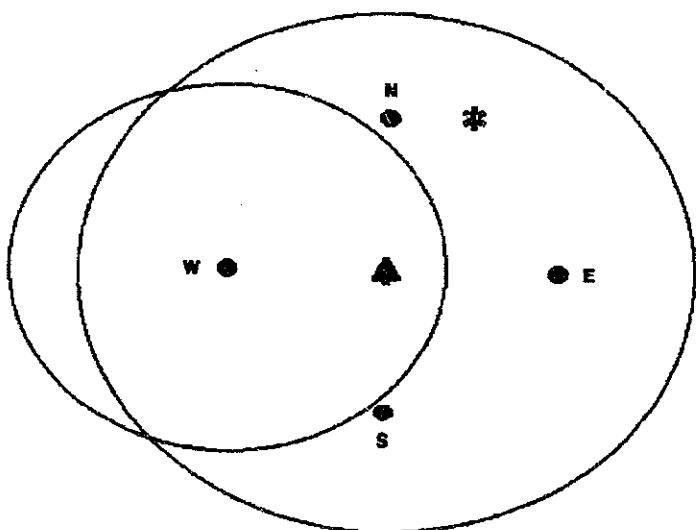


FIGURE 6.1. A spatial environment consisting of a central landmark (shown as a tree) surrounded by four other landmarks (shown as disks). Each landmark possesses a distinctive "odor" which can be sensed at a distance. Odor distributions decrease linearly from their associated landmarks and become undetectable at the large ellipses. The asterisk shows the location of the ASN.

giving the odor concentrations due to the central "tree" and the north, south, east, and west neutral landmarks.

Figure 6.2 shows an ASN with 5 input pathways, labeled vertically on the left according to the landmarks to which they respond. The shaded input pathway N indicates that the ASN is near the north neutral landmark. There are 4 output pathways labeled horizontally at the bottom as controlling "actions." The manner in which these actions determine locomotion was chosen solely for the sake of simplicity. There is an output element for each compass direction. Each output element produces an output of 0 or 1 at each time step. For example, if $N=0$, $S=1$, $E=1$, and $W=0$ (as shown by the shaded output elements in Figure 6.2), the ASN will move a fixed distance south and east. We use a kind of "reciprocal inhibition" between the north and south elements and between the east and west elements so that at each time step usually only one of each pair of elements outputs a 1. Clearly, we are not attempting to model in any detailed manner the motor control system of an organism (for example, there is no explicit spatial orientation of the ASN).

The arrangement of input and output pathways used in Figure 6.2 permits the connection weights to be displayed in convenient form as circles centered on the intersections of input pathways and the vertical output element "dendrites."

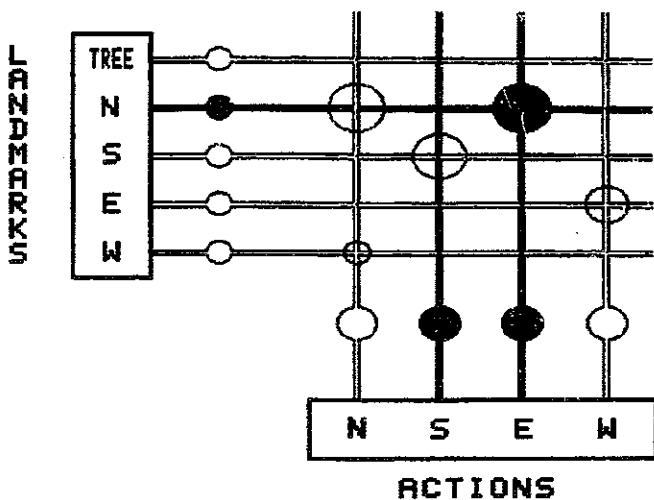


FIGURE 6.2. The ASN controlling locomotion in the spatial environment. The five input pathways are labeled vertically on the left according to the landmarks to which they respond. The shaded input pathway N indicates that the ASN is near the north neutral landmark. The four output pathways controlling actions are labeled horizontally at the bottom according to the direction of movement they cause. The shaded output elements indicate that a southeast movement is being made. The associative matrix weights are displayed as circles centered on the intersections of the horizontal input pathways and vertical output pathways. Positive weights are shown as hollow circles, and negative weights are shown as solid circles.

Positive weights are shown as hollow circles, and negative weights are shown as solid circles. The sizes of the circles indicate the relative magnitudes of the corresponding weights. The uppermost "tree" input is the specialized payoff pathway z which has no associated weights. These connection weights form an associative matrix which is similar to those widely discussed in the literature (e.g., Anderson et al., 1977; Amari, 1977a, b; Kohonen, 1977) but one that gathers information by means of the more complex closed-loop learning rules to be described.

The ASN's task in this environment is to 1) find the central landmark by climbing the attractant distribution and 2) associate with each place that action which causes movement toward the central landmark. The first part of this task is a simple hill-climbing problem that does not require long-term memory. The second part is an example of the associative search task. Although the payoff signal is derived from a single spatial distribution (the "odor" of the tree), the optimal action is clearly a function of the ASN's location. For example, if the ASN is south of the central landmark, it is best for it to move north; if it is north of the central landmark, it is best for it to move south. Consequently, the search for the optimal action in each place requires maximization of functions of ASN actions which differ from place to place. (A predictor as discussed

in Section 5.4 is not required for this spatial learning task since the functions to be maximized vary smoothly over time.) As a result of solving the second part of this problem, the ASN can proceed directly to the central landmark simply by performing the actions associated with its successive locations. Importantly, this direct approach is possible when the attractant distribution is very noisy, intermittent, or even totally absent (as we demonstrate below).

6.4 The Learning Rule

The ASN presented here uses the same type of learning rule as discussed in Section 5. Let $x_1(t)$, $x_2(t)$, $x_3(t)$, and $x_4(t)$ denote the signals at time t from the north, south, east, and west landmarks respectively, and let $z(t)$ denote the signal from the central landmark. Each output element j , $j = 1, \dots, 4$, has a weight w_{ij} associated with neutral landmark input x_i , $i = 1, \dots, 4$, and an additional weight w_{0j} . Let $w_{ij}(t)$, $i = 0, \dots, 4$, denote the values of these weights at time t . Let

$$s_j(t) = w_{0j}(t) + \sum_{i=1}^n w_{ij}(t)x_i(t).$$

The output of element j at time t is

$$y_j(t) = \begin{cases} 1 & \text{if } s_j(t) + \text{NOISE}_j(t) > 0 \\ 0 & \text{otherwise} \end{cases} \quad (6.1)$$

where each NOISE_j , $j = 1, \dots, 4$, is a mean zero normally distributed random variable (with the same variance for each j).

At each time step, each weight w_{ij} , $i, j = 1, \dots, 4$, is updated according to the following equation:

$$w_{ij}(t+1) = w_{ij}(t) + c[z(t) - z(t-1)]y(t-1)x_i(t-1). \quad (6.2)$$

The weights w_{0j} are updated as follows:

$$w_{0j}(t+1) = f[w_{0j}(t) + c_0(z(t) - z(t-1))y(t-1)] \quad (6.3)$$

where

$$f(x) = \begin{cases} \text{BOUND if } x > \text{BOUND} \\ 0 \text{ if } x < 0 \\ x \text{ otherwise} \end{cases}$$

bounds each w_{0j} to the interval $[0, \text{BOUND}]$. The parameters c and c_0 are positive real numbers determining rates of learning. In all of the simulations described below, $c = 0.25$, $c_0 = 0.5$, $\text{BOUND} = 0.005$, and the standard deviation of the random variable NOISE_j was 0.01 for $j=1, \dots, 4$.

Equation 6.2 implies that if the firing of an output element in a given place is followed by a movement toward higher attractant concentration z , then the element will become more likely to fire in that place in the future. If firing is followed by a movement toward lower values of z ,

firing will become less likely in that place. See Section 5 for a more detailed discussion of this class of learning rules [footnote].

The weights w_{0j} changing according to Equation 6.3 permit the ASN to climb the attractant distribution in the absence of landmark information. Equation 6.3 is similar to Equation 6.2 applied to a constant signal from a universally present landmark ($x_0(t) = 1$ for all t). If c_0 is sufficiently large compared to BOUND (as it was in our simulations), then complete learning will occur in a single trial so that a movement in an up-gradient direction will tend to be followed by a movement in the same direction. This straight line trajectory will tend to continue until it takes the ASN down-gradient. Down-gradient moves will drive w_{0j} to zero so that the random component will dominate. The bound function f is necessary to insure that down-gradient moves can return the weight to zero. The resulting hill-climbing strategy is similar to that used by certain types of bacteria to climb nutrient gradients (Koshland, 1973). Fraenkel and Gunn (1962) call this strategy

Equation 6.2 is identical to Equation 5.2 except that the term $y(t-1)$ is used here instead of $y(t-1) - y(t-2)$. In the experiments of Section 5, changes in z were attributable to changes in y . Here, y itself determines the change in z because it causes a change in spatial location rather than movement to a particular place.

Klino-Kinesis and Selfridge (1978) calls it "Run and Twiddle" (if things are improving, keep doing what you are doing; if things get worse, do something else).

6.5 Learning in a Noiseless Environment

If the attractant concentration can be reliably sensed, then the hill-climbing part of the ASN's task can be accomplished easily. Figure 6.3 shows the ASN's trajectory for the case in which there are no neutral landmarks. The central landmark is approached due to the action of Equation 6.3. Since no associations are formed in this case, that is, since no long-term memory traces are formed, later attempts to climb the same hill will proceed at essentially the same rate as the first attempt.

Figure 6.4 illustrates the ASN behavior in the presence of the neutral landmarks. Figure 6.4A1 shows the ASN behavior for 35 time steps. Figure 6.4A2 shows the state of the ASN as a result of this behavior. Nonzero weights have appeared associated with the north and east landmark input pathways since the ASN has remained in the vicinity of these landmarks (and hence only these pathways were eligible for modification). Since movements north and south were correlated respectively with decreases and increases in the

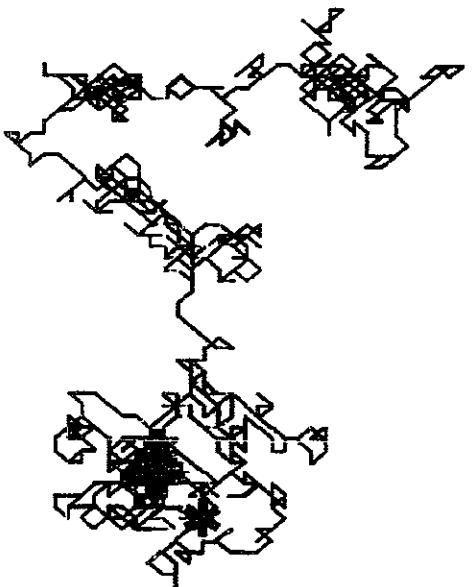


FIGURE 6.3. The ASN's path is shown as it climbs the attractant gradient in the absence of landmark guidance. No long-term memory traces are formed, and later attempts to climb the same gradient will proceed at essentially the same rate.

attractant level, weights have formed so that the north and east landmark "odors" inhibit movement north and excite movement south. Weights associated with the east landmark pathway are smaller in magnitude than those for the north landmark since the ASN remained closer to the north landmark. Similarly, the north and east landmark inputs inhibit movement west. Weights for the east output element are too small to be visible since the ASN only infrequently moved east.

Figure 6.4A3 shows the results of learning in a vivid form. A vector is shown at each point in a grid covering the entire space. Each vector is the result of computing the values s_j , $j = 1, \dots, 4$, from the ASN input vector associated with the place at which the vector appears. The resulting 4-tuple is displayed as a vector in the obvious way. The direction of the vector at each location gives the direction of the ASN's most probable first step if it were to start at that location. The vector's magnitude is related to the probability that the ASN will take this step. It is important to note that the attractant distribution of the central landmark is not used to determine the vector fields. The vectors represent information stored in the ASN's memory; not information directly present in the environment. The vectors show how the ASN would tend to move even if the central landmark and its attractant

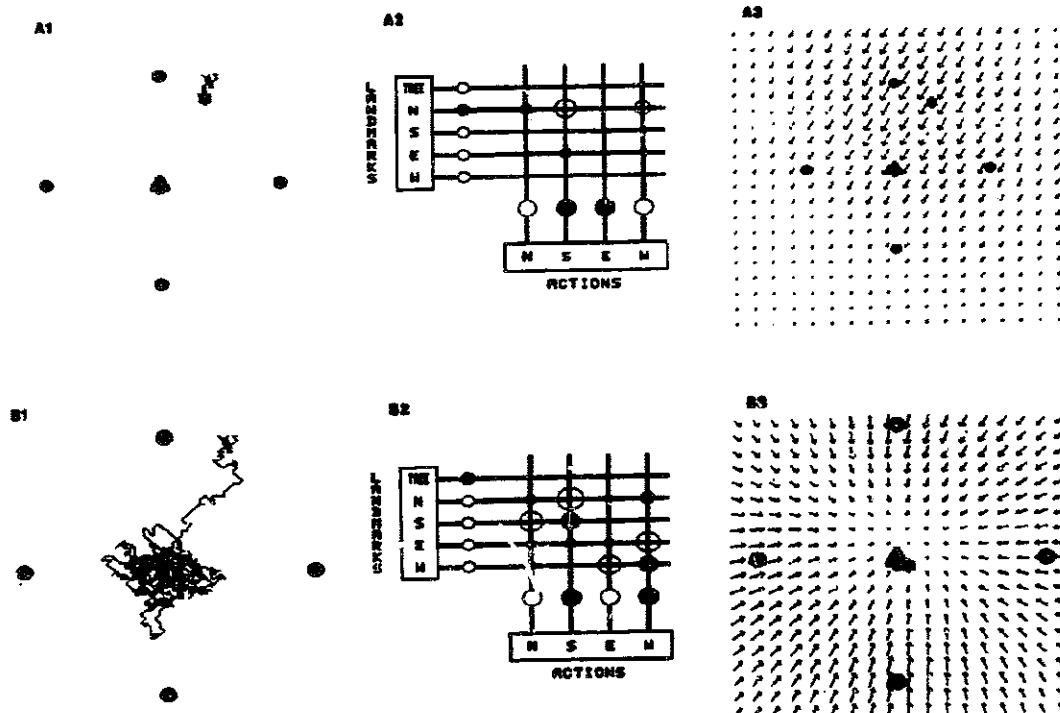


FIGURE 6.4. ASN behavior in the presence of neutral landmarks. A1) ASN behavior for 35 time steps. A2) The state of the ASN as a result of the experience shown in A1. A3) A vector field representation of the ASN state shown in A2. B1) ASN behavior for about 800 time steps. B2) The state of the ASN after about 800 time steps shows that proximity to the north landmark will make the ASN move south, proximity to the south landmark will make it move north, and similarly for the east and west landmarks. B3) A vector field representation of the ASN state shown in B2.

distribution were not present. The generalization capability of the ASN is clearly shown by the vectors associated with places never visited by the ASN.

Figure 6.4B shows how the ASN behaves for about 300 time steps. It climbs the attractant distribution and remains in the vicinity of the central landmark (Figure 6.4B1). The resultant associative matrix values (Figure 6.4B2) show that the north landmark signal inhibits the north output element and excites the south output element. Consequently, when the ASN is in the vicinity of the north landmark, it will tend to move south. Similarly, a strong signal from the south landmark will cause the ASN to move north. The weights associated with the east and west landmarks similarly affect the east and west output elements. The resultant movement tendencies are shown as a vector field in Figure 6.4B3. This form of learning is not dependent on the central location of the attracting landmark. Figure 6.5 shows a vector field determined from the contents of the ASN's memory after about 300 time steps of learning with the attracting landmark located off center. The importance of this illustration is that it shows that the learning rule is capable of not only determining the correct signs for the weights but also their correct magnitudes.

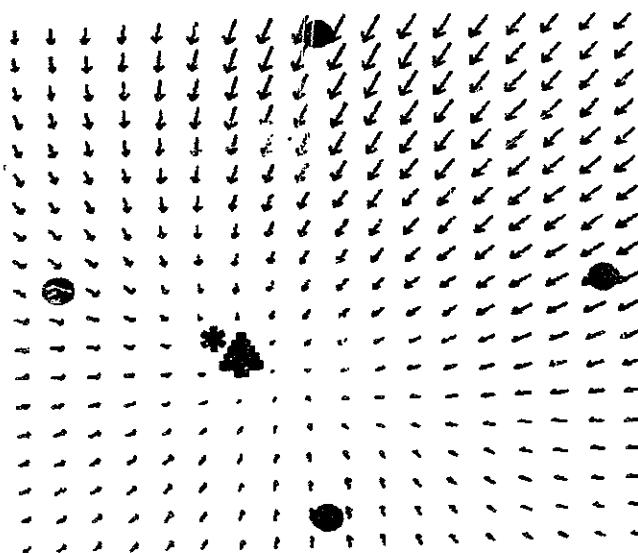


FIGURE 6.5. A vector field representation of the ASN's state after about 800 time steps in an environment with the attractant landmark located off center. The learning rule is capable of determining the correct magnitudes for the weights in addition to the correct signs.



FIGURE 6.6. Use of long-term memory. With the ASN state as shown in Figure 6.4B2 and the central landmark and its attractant gradient removed, the ASN takes a direct route to the central landmark's former position from a place it has never before visited. Stimulus patterns associated with successive positions "key-out" the appropriate actions.

The information stored in the association matrix formed during exploration of this spatial environment can be used by the ASN to guide movement even in the absence of the attractant gradient. In Figure 6.6 is shown the behavior of the ASN after learning by exploration of the environment with the attractant landmark in the center. The central landmark and its attractant distribution have been removed from the environment, and the ASN starts at a place it has never before visited. The ASN takes a direct route to the former location of the central landmark. This occurs because the input vector associated with each place "keys out" the appropriate action. The ASN remains near the central landmark's former location.

6.6 Relearning in a Modified Environment

Here we illustrate how the ASN can reorganize its associative matrix due to changes in its environment. We allowed the ASN to learn in the original environment (Figure 6.1) until it was able to associate the best movement with each place. We then interchanged the east and west landmarks. Figure 6.7A shows the vector field resulting from evaluating the ASN's associative matrix in the altered environment. The central landmark location is now a saddle point rather than a stable focus. Starting from a central

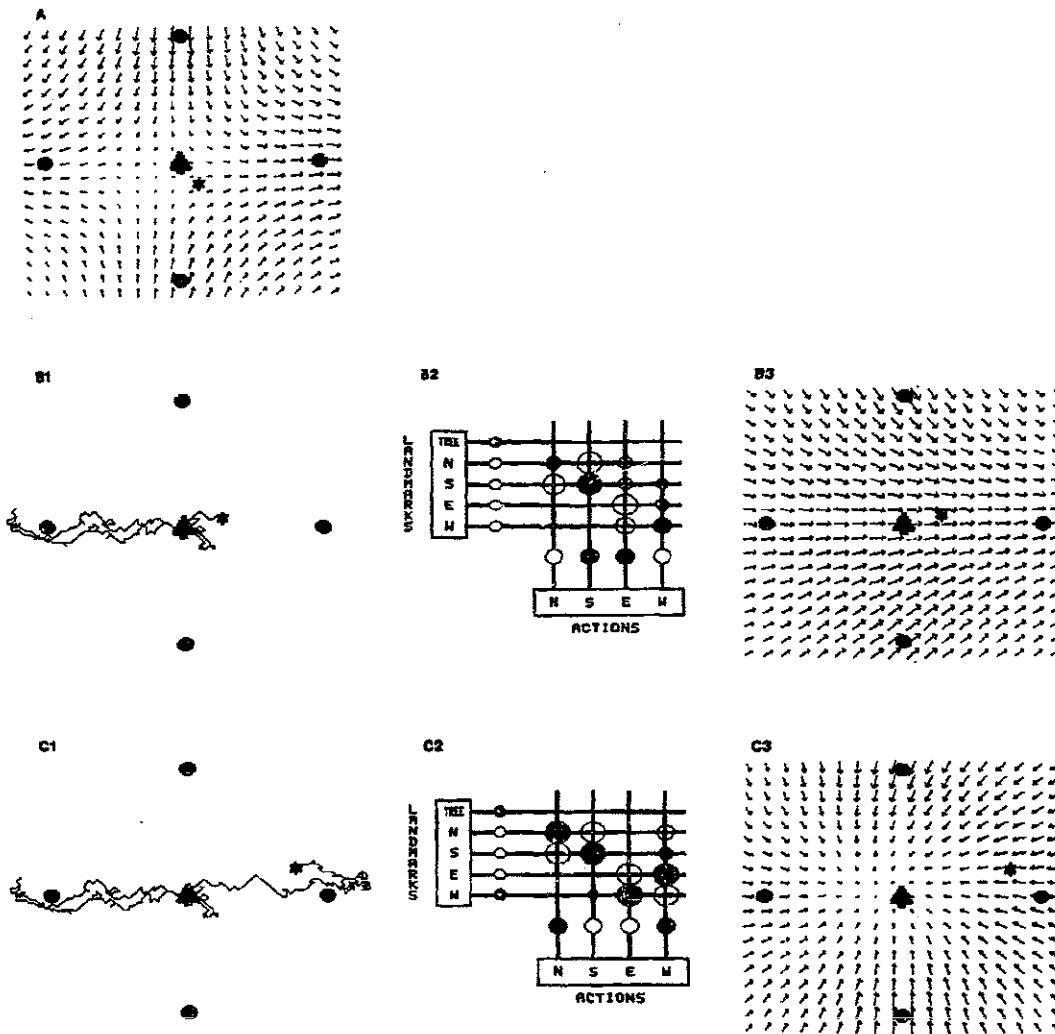


FIGURE 6.7. Relearning in a modified environment. After learning in the original environment (Figure 6.1) we interchanged the east and west landmarks. A) The vector field resulting from evaluating the ASN's state in the altered environment. B1) The spatial path of a western excursion. B2) The ASN state after the excursion west shown in B1. B3) The vector field representation of the ASN's memory contents after the excursion west shown in B1. C1) The spatial path of an eastern excursion. C2) The ASN state after the eastern excursion. C3) The vector field representation of the ASN's memory contents after the excursion east shown in C1.

position, the ASN is "misled" by its sensory information and follows the vector field away from the central landmark (Figure 6.7B1). Since this movement is down the attractant gradient, the ASN alters the weights to the east and west output elements from the east neutral landmark input (which now responds to the landmark to the west). This relearning results in the network of Figure 6.7B2 and the vector field of Figure 6.7B3. A similar excursion to the east modifies the weights associated with the west neutral input which now responds to the landmark to the east (Figure 6.7C). If the attractant distribution had been absent, no relearning would have occurred.

6.7 Learning in a Noisy Environment

Climbing a hill as large and reliably sensed as the attractant distribution of the preceding illustrations is not a difficult task. When the attractant concentration can be sensed only in the presence of noise, the task becomes more difficult and more interesting. The sensitivity of the ASN to neutral context information permits it to improve its performance in climbing a noisy hill with repeated attempts [footnote]. Figure 6.8A shows the ASN performance, starting

Although we do not illustrate it here, we would expect that context information would also facilitate the more difficult problem of higher dimensional search.

with all weights zero, as it climbs the attractant concentration corrupted by additive noise. The noise is normally distributed with a standard deviation of 0.02. Comparing Figure 6.8A with Figure 6.3 or Figure 6.4B1 shows that hill-climbing performance is significantly degraded. After sufficient experience with the noisy attractant concentration (1107 time steps), the ASN uses neutral landmark guidance to directly approach the goal even with the same noise level in the attractant concentration (Figure 6.3B).

There are other means for improving hill-climbing performance in the presence of noise such as direct low-pass temporal filtering of the attractant signal as it is received by the ASN over time. We have not optimized hill-climbing behavior of the ASN in the absence of landmark guidance. Consequently, Figure 6.8 does not compare landmark guided hill-climbing with the best hill-climbing behavior that can be accomplished without landmark guidance. What is important in this comparison, however, is that the association of neutral context information during a search permits the system to improve its performance with repeated attempts to approach a goal in the same or similar environments. Even the most highly tuned pure hill-climbing strategy does not learn from its experience in this manner.



FIGURE 6.8. Learning in a noisy environment. A) ASN behavior, starting with all weights zero, as it climbs the attractant gradient corrupted by additive noise. Hill-climbing performance is significantly degraded (cf. Figure 6.3 or Figure 6.4B1). B) After sufficient experience with the noisy attractant gradient (1107 time steps), the ASN uses neutral landmark guidance to directly approach the goal even with the same noise level in the attractant gradient. Previous experience in the same or similar environments can be used to improve performance.

This example illustrates that the exploitation of sensory context can provide significant adaptive advantages if the same or similar search problems occur repeatedly.

6.8 A Remark on Linearity

The associative search problem posed by the spatial environment of Figure 6.1 is simple enough to be solvable by an ASN capable of making only linear associations. The influences of the neutral landmarks merely superimpose to form the desired control surface. If this were not the case, the ASN which we have described would not be able to form a stable mapping. Due to its linearity, it is not able to represent arbitrary patterns of location-action associations; that is, only certain types of vector fields can be learned.

In our current research, we are investigating two methods for extending the ASN's capabilities to include nonlinear associations. The first relies on the observation that more varied associations can be formed as the number of landmarks increases. If, for example, there were a distinguishable landmark at each spatial location, then a linear ASN could learn arbitrary location-action associations (this would be similar to the approach taken in

the BOXES system of Michie and Chambers, 1968). This suggests that it would be useful for a system to effectively "create" landmarks where needed in order to refine its representation of space. Such a landmark, which we call a "virtual landmark," would be created by the formation of an appropriate nonlinear combination of the sensory signals provided by the real landmarks. Another approach to nonlinearity is related to the "Patchwork Map" theory described by Kuipers (1977). Here, the system's knowledge of space would consist of several different associative mappings appropriate for guiding behavior in different regions of space. The system would need to develop nonlinear switching capabilities for accessing the correct associative structure when entering each region. Both of these approaches to nonlinear learning are applicable to a wide variety of spatial and non-spatial problems. We are finding that the simple spatial interpretation described in this section provides a concrete and generalizable framework for approaching these very difficult and general problems.

6.9 Conclusion

We have illustrated the behavior of an ASN in a simple spatial learning task. The spatial problem provides a vivid way to demonstrate the search, association, and generalization capabilities of an ASN. Although we have

illustrated these capabilities in an extremely simple form, it should be realized that the methods employed have much wider applicability. The spatial learning problem is an example of a wide class of problems, some of which require paths to be learned through spaces which do not necessarily represent physical space. For example, the space may be the state space of a dynamical system, in which case the vector fields developed represent hypothesized system dynamics. Associative learning capabilities provide a simple means whereby experience in attempting to solve a problem can be accumulated and used to drastically improve performance in similar problems. The necessity for explicit search is minimized by storing in long-term memory the information gained in previous searches.

Finally, we wish to comment on the simplicity of the ASN illustrated. It consists of just four adaptive elements acting in parallel. Since the adaptive elements themselves embody fairly sophisticated learning rules, utilizing both short-term and long-term memory, we did not need to construct a special purpose network to perform the landmark learning tasks which we have presented. The behavior illustrated is a very natural consequence of a set of elements operating according to a carefully designed closed-loop learning rule.

SECTION 7

AN ADAPTIVE NETWORK THAT CONSTRUCTS AND USES AN INTERNAL MODEL OF ITS WORLD

7.1 Internal Models for Search and Simulation

The words "internal model of the world" have been used by many theorists of the mind to refer to some kind of store of knowledge within an adaptive system that it uses to better interact with its world (e.g., Arbib, 1972; Craik, 1943; Gregory, 1969, MacKay, 1955; Piaget, 1954). The ideas behind these models vary from the idea of a very general knowledge store capable of answering any sort of question about the world, to extremely limited knowledge stores that can answer only a single question: What should be done next? The kind of internal model we are concerned with in this section is of a generality intermediate between these two extremes. By an internal model we will mean any part of an adaptive system which can provide expectations or

predictions about what would happen in particular situations. Further, we are concerned specifically with those cases in which the model is used to mentally simulate the consequences of various actions in order to choose among them without having to try them overtly. The following few pages focus and expand upon this idea.

Kenneth Craik (1943) was one of the first to clearly state the view of thought as an internal simulation of the world, allowing many courses of action to be hypothetically attempted and evaluated:

If the organism carries a "small-scale model" of external reality and of its own possible actions within its head, it is able to try out various alternatives, conclude which is the best of them, react to future situations before they arise, utilize the knowledge of past events in dealing with the present and future, and in every way to react in a much fuller, safer, and more competent manner to the emergencies that face it. (p.61)

Aspects of this theory of thought, however, are much older than Craik's work. Donald Campbell (1962) traces a very similar theory of "creative thought" back to the writings of Alexander Bain (1855, 1874), Ernst Mach (1896), and Poincaré (1908, 1913).

Campbell (1962) emphasizes that the interaction with both the world and the internal model can involve trial and error:

At this level [the level of creative thought] there is a substitute exploration of a substitute representation of the environment, the "solution" being selected from the multifarious exploratory thought-trials according to a criterion substituting for an external state of affairs. In so far as the three substitutions are accurate, the solutions when put into overt locomotion are adaptive, leading to behavior which lacks blind floundering... (p.212-3)

Unfortunately, the idea of "trial and error" in search has frequently been mistaken for that of random or blind search. A search by trial and error can be a highly structured and heuristically guided one. By trial and error search we mean any search undertaken under the guidance of a certain kind of feedback process in which options are tried and then evaluated and retracted or changed if in error. Any "hypothesis and test" search, or any search using backtracking, would qualify as a search using trial and error in this sense.

Internal trial and error as a model of thought and reasoning turns out to be a view that is held extremely widely among theorists of the mind. Such a modeling/simulation view plays an important role in the theories of Dennett (1978) in philosophy; Simon (1969) in artificial intelligence; Sommerhoff (1975) and Arbib (1972) in brain theory; Dawkins (1976) in biology; Galanter and Gerstenhaber (1956) and Miller, Galanter and Pribram (1960) in psychology; to name just a few. Figure 7.1 summarizes the essential features of this view of thought as used in

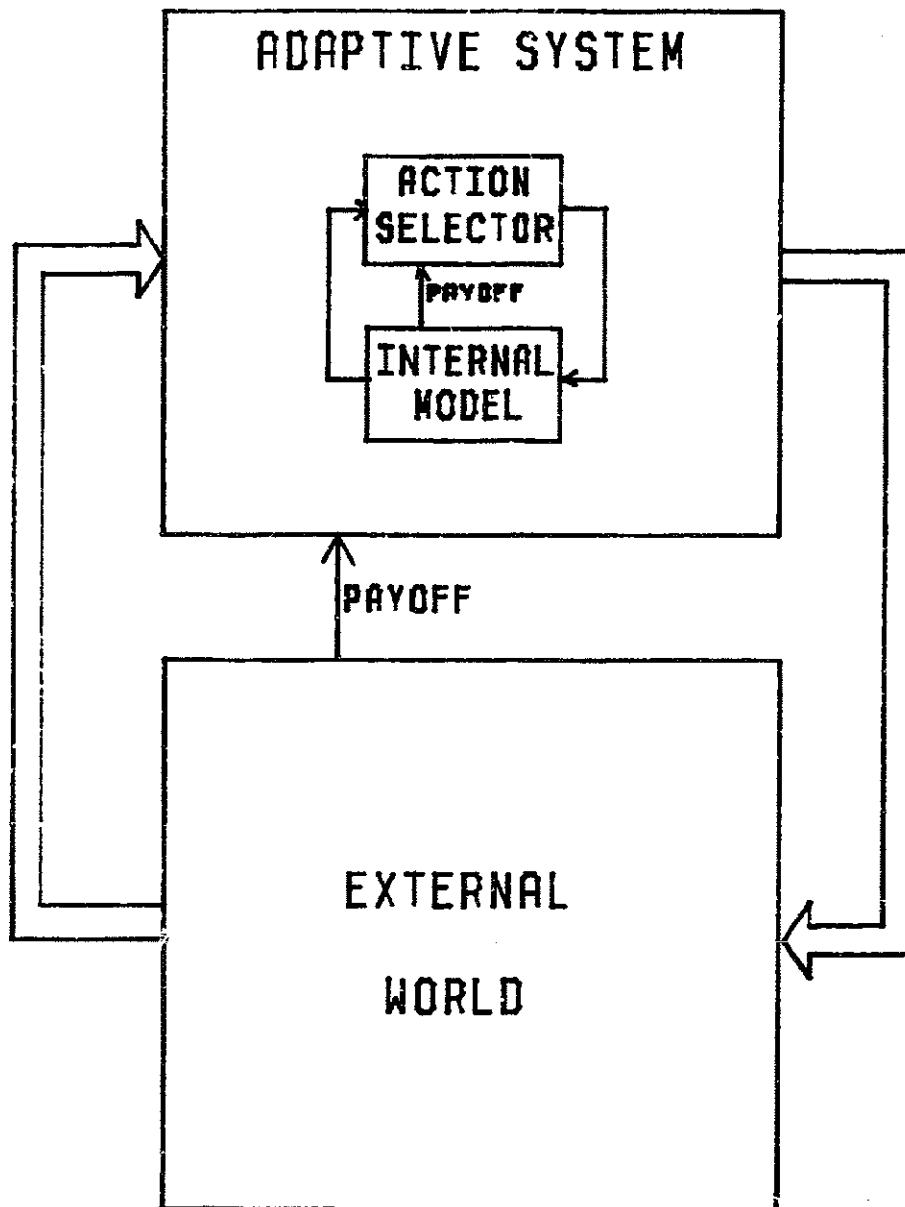


Figure 7.1. An adaptive system based on the idea of internal simulation. The system interacts with its model in the same way that it interacts with the real world.

this section: An organism constructs an internal model of the world that allows prediction of the observable behavior of the world as a function of possible actions by the organism. The internal model is used to select behavior in an interactive manner identical to the interaction with the external environment in the absence of a model. Trial and error search for the action which achieves the best result from the external environment is replaced by covert, internal trial and error search for the hypothetical action which secures the best anticipated result from the internal model. The internal model must be either faster, easier, or safer to interact with than the external environment in order for it to be useful.

This section takes a few first steps towards formalizing this model-based theory of thought. The animal learning theory literature has been found to be extremely useful in obtaining a more concrete idea of what it means to create and use an internal model of the world. Since the concept of an internal model is, by definition, a mediating theoretical construct not directly associated with overt behavior, psychologists have concentrated on devising experiments which we can view as revealing indirect effects of the model on behavior. These animal learning theorists called the phenomena their experiments revealed such things as reasoning, latent learning, and insight. The centerpiece

of this section is the presentation of a completely defined adaptive network which constructs and uses an internal model to solve a task similar to one in the animal learning theory literature. Both the adaptive network and the task environment were simulated by computer. The intent was to find as simple a network and task as possible while still being able to demonstrate behavior that psychologists would consider "reasoning," or model-requiring.

Figure 7.2 is a floor plan of an early form of a classic maze problem for rats (Tolman and Honzik, 1930). Its solution is considered to involve spatial reasoning capabilities. To oversimplify, the rats were familiar with all three paths to the goal, and preferred them in order of increasing path length: A over B, and B over C. When a block was introduced as shown, the rats tried A, discovered the block and then predominantly chose path C, the longest of the paths, next. Since their normal preference when A is blocked was B, the path of intermediate length, this result indicated that the rats used some sort of spatial map, or model of the maze, which informed them that path B was also blocked. This experiment was seen as a positive test of insight or reasoning in the rat.

A much simpler experiment of the same intent uses a one-choice T-maze with detachable distinguishable goal boxes

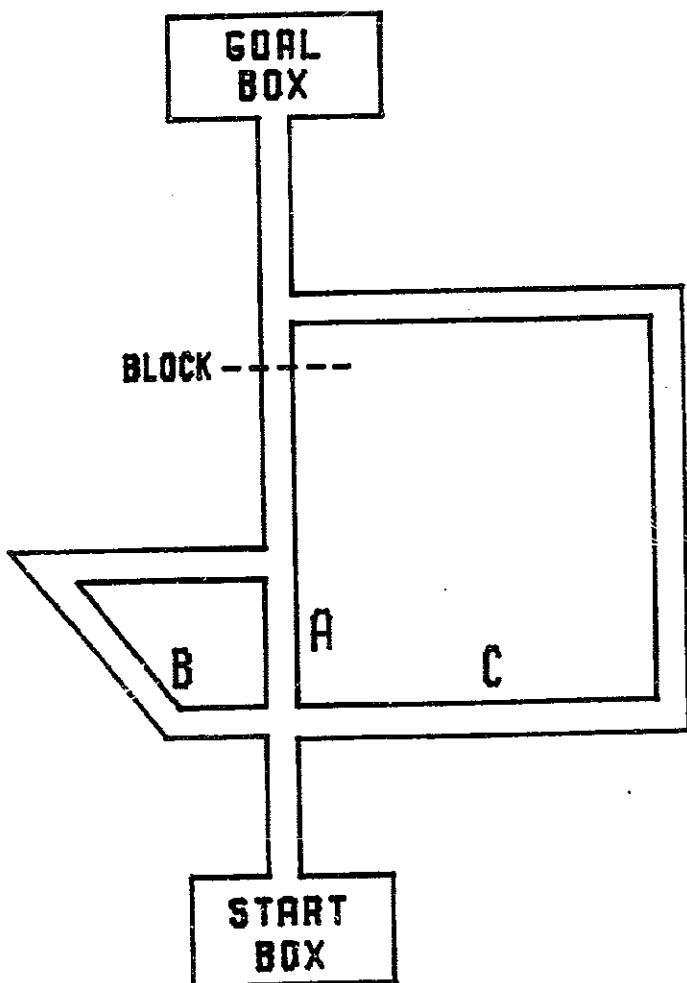


Figure 7.2. A maze used to test insight in rats. The rats are familiar with all three paths to the goal box and prefer them in order of decreasing length: A over B, B over C. If they have "insight," then after taking A to discover the block, they next try path C rather than B.

141

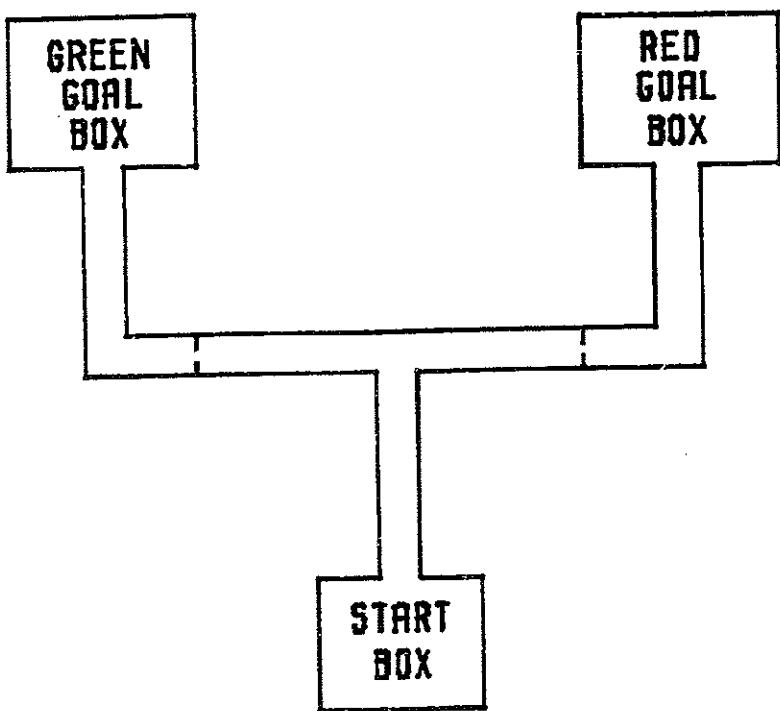


Figure 7.3. A simple T-maze task with distinguishable detachable goal boxes used to test latent learning and reasoning in rats. The rat cannot see or backtrack through the one-way doors indicated by dashed lines. This task is conceptually very similar to the one posed to the simulated adaptive network presented in this paper.

(Figure 7.3). Since this problem is very similar to the one we have posed to the simulated adaptive network, we will describe it in greater detail. There are three phases to the experiment: In the exploration phase the subject is repeatedly placed at the entrance to the maze. When the subject reaches one of the goal boxes, it is removed from the apparatus. There is no food or other reinforcer anywhere in the maze. Backtracking is not allowed. In the association phase the goal boxes are separated from the T-maze and carried to another room. There the subject is fed in the red goal box that was on its right, and given a painful electric shock in the green goal box that was on its left. In the testing phase the subject is returned to the start of the T-maze.

The key question is: Which way will the subject turn on the first post-training trial? Most rats will turn right. Note that neither the action of turning right nor the action of turning left is ever temporally associated with reward or punishment in this experiment. In order to solve this task, the subject has to combine two separately learned facts about the world: 1) that turning right in the T-maze will bring it to the red goal box and turning left will bring it to the green goal box, and 2) that the red goal box is a place where it may be fed, and the green goal box a place where it may be shocked. It is this combination

which is thought of as the reasoning process, a sort of transitivity of prediction or primitive modus ponens.

Viewing the solution of this T-maze problem as an instance of the use of an internal model, in this case a spatial cognitive map, suggests two aspects of the idea of simulation by internal model that may account for the popularity and apparent promise of the idea. First, the sort of reasoning by predictive transitivity mentioned above is precisely the sort of reasoning that is achieved by a simulation. To simulate a complex system by computer, we provide the step-by-step transition dynamics of the system, and the simulation scheme repeatedly applies these dynamics to update the state of the simulated model. In just this way a simulation can combine "right turn predicts (arrival at) red goal box" and "red goal box predicts food" to infer that food can be attained by turning right. Such a capability for propagating predictions is an important component of the ability to generate the consequences of proposed actions.

The second important aspect of the idea of simulation by internal model that appears in this simple T-maze example is that it provides a framework for learning about the environment even in the absence of rewarding or punishing events. For example, forming an internal model becomes the

purpose and explanation for the T-maze subjects' learning that turning right leads to a red area, even though no reinforcing events occur. The problem of learning about the determinants of all stimuli is much more difficult than that of merely learning about the determinants of a few designated reinforcing stimuli. This turns out to be an important problem for adaptive network research as well as animal learning theory. Early attempts in both these fields (e.g., Thorndike, 1911; Clark and Farley, 1955) used reinforcement to form associations between stimuli and responses. However, as was learned from the latent learning experiments (Blodgett, 1929), animals do learn in the absence of reinforcing events of any kind.

Reinforcement, being a one-dimensional measure, provides very little information compared to the torrent of sensory information available. It has become generally recognized that intelligent artificial adaptive systems also must use this additional information (see the discussion of the "apportionment of credit problem" in Minsky (1961)). Much of the promise of the idea of an internal model may be that this concept explicitly encourages and provides a way of understanding learning in the absence of reward or punishment. This type of learning involves the construction of an accurate predictive model, a process that is normally independent of reinforcement. Once the model is formed,

internal trial and error through simulation provides a framework for using the information picked up from the environment.

7.2 The Simulated Task Environment

In the example that we implemented, the experimental design, the environment, the experimental subjects, and the adaptive networks to control them, could all be selected for our convenience. This allowed further simplification in the design of the reasoning task. The ground plan of the environment is shown in Figure 7.4. The lower area is used in a manner analogous to the T-maze, the two regions on the right and left being analogous to the red and green goal boxes at the ends of the T-maze. The two enclosed regions shown in the upper part of Figure 7.4 are analogous to these same goal boxes when they have been moved to another room for association with food and shock in the absence of the T-maze. That these are actually separate regions is of no importance here: The adaptive networks controlling the simulated beasts have only three sensory input lines, one for sensing being within a green region, one for sensing being within a red region, and one for sensing rewarding stimulation. In terms of this limited sensory vocabulary,

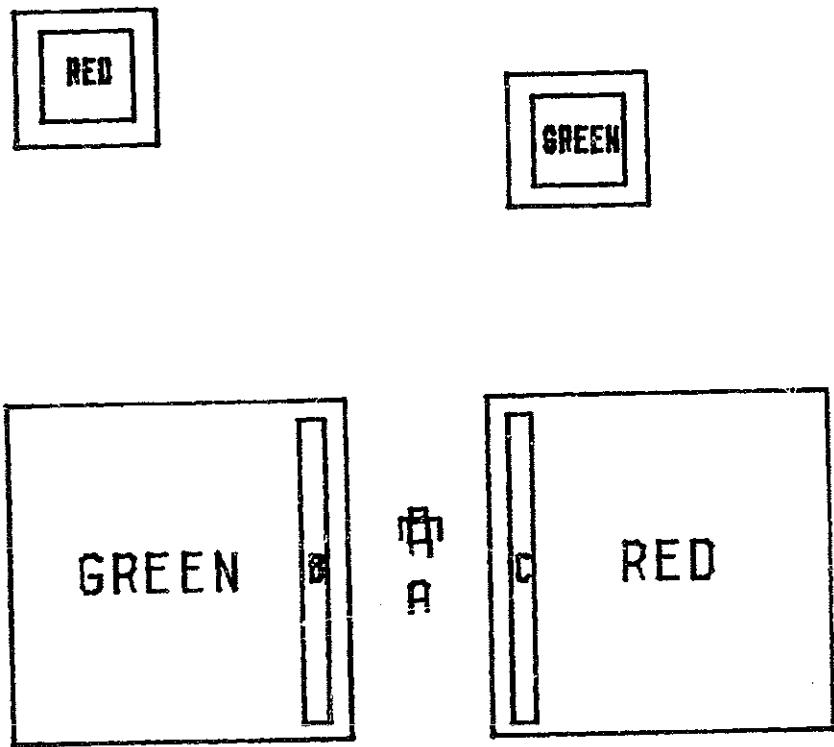


Figure 7.4. Ground plan of the simulated environment. The lower area is used in a manner analogous to the T-maze in Figure 7.3, the two regions on the right and left being analogous to red and green goal boxes at the ends of the T-maze. The upper two enclosed regions are analogous to those same goal boxes when they have been moved to another room for association with food and shock in the absence of the T-maze.

all regions of the same color are indistinguishable. This is clearly an enormous simplification of the perceptual process.

The simulated beasts have only two graded actions: move to the right, move to the left. These are meant to be extreme simplifications of, and yet analogous to, the right-turn and left-turn actions of the T-maze task. In the exploration phase of the simulation experiment, the beasts are placed at A, between the two large colored regions (Figure 7.4), and allowed to wander back and forth randomly. The barriers at B and C obstruct their movement thereby preventing them from moving too far away. This insures that they eventually gain experience moving to and from both regions. Thus, all trajectories are along a straight horizontal line between the two barriers. The two upper goal box areas shown in the upper part of Figure 7.4 are used in the association phase. For the testing phase, the beasts are returned to location A between the lower two regions to see which region is entered first.

We next describe the adaptive network and then proceed through each phase of the simulation experiment, discussing the experimental manipulations and network changes in detail. For reference, Appendix D contains a summary of the details of the three phases of the simulation experiment,

and Appendix E contains a detailed specification of the simulated adaptive network model.

7.3 The Simulated Adaptive Network

Figure 7.5 is a block diagram of the adaptive network design. The network is divided into two major components: An action selecting mechanism and an internal model of the environment. The action selection mechanism uses the actual environment and the model of the environment in exactly the same way - both provide feedback to evaluate actions attempted by the action selecting mechanism. The evaluations by the model and by the environment of the most recently selected action are added to yield the evaluation input to the action selecting component. Importantly, the feedback loop through the internal model is much faster than the feedback loop through the environment, and thus proposed actions can be evaluated by the model so quickly that the rejected alternatives have very little influence on the environment and the organism's overt behavior. This is accomplished in the simulated example system by letting the motion of the simulated beasts depend not only on the instantaneous action selected, but also on past values, in an exponentially weighed manner. The result is that even

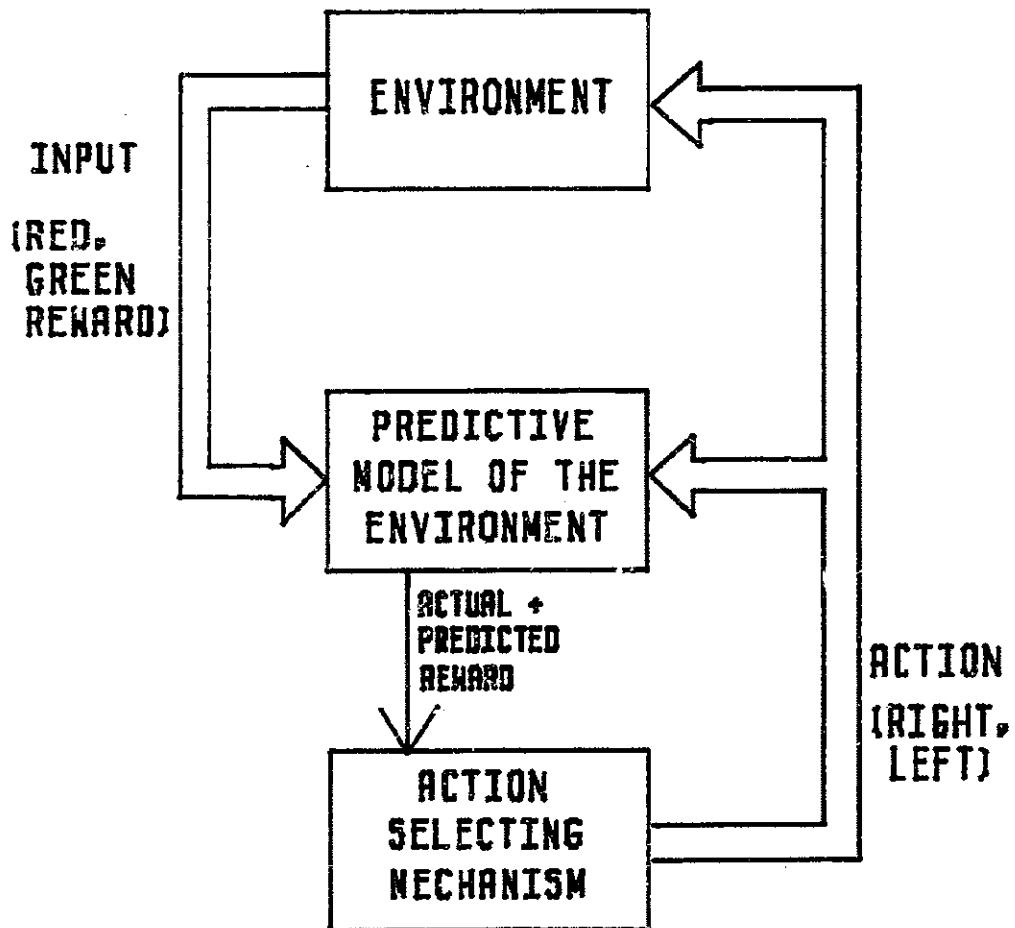


Figure 7.5. Block diagram of the adaptive network and its connection to the environment. The action selecting mechanism has its choices evaluated via two feedback loops; one through the environment, and one through an internal model of the environment. If the model is faster than the environment, then the feedback loop through the model will control overt behavior.

though both the action selection and the overt action are changed and updated every time step of the simulation, the environment is slightly "viscous" relative to the network dynamics, acting as a "leaky integrator," or a system with inertia that must be overcome before overt action aligns with the current action selection. A decisive overt movement only occurs once the system has converged onto a particular choice of action. Maintaining a particular action as the one selected for a significant period of time (about four time steps in the simulated system) causes that action to become expressed in overt movement.

7.3.1 The Action Selecting Component

The division of the adaptive network into the action selecting and internal model components makes its construction from adaptive elements relatively simple. All that is needed for the action selector is a bank of action elements which correlate their output, or action, with increases or decreases in the evaluation or reinforcement input to this subsystem (Figure 7.6). We will need one element whose action represents the tendency to turn right and one whose action represents the tendency to turn left. The environment will then resolve any conflict between these

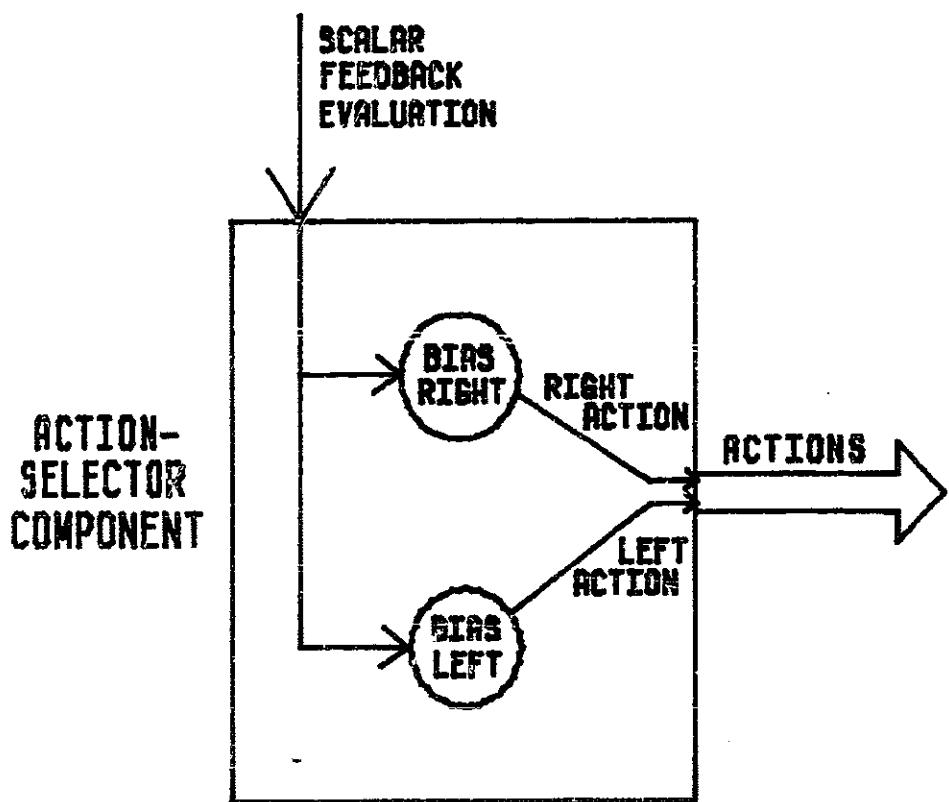


Figure 7.6. Detail of the action-selector component of the simulated network blocked out in Figure 7.5. The elements correlate their output, or action, with increases or decreases in the evaluation input.

two by responding to the difference in the tendencies, much as our skeletons resolve the conflict between agonist and antagonist muscles (this is explained in detail in Appendix D). The action levels are originally chosen randomly, but if a correlation is found between action level and subsequent evaluation, the choice of action is biased to make positively correlated actions positively correlated more likely to be selected and negatively correlated actions less likely. Mathematically, the momentary action choice of each element is the sum of a random component and a bias or accumulated correlation component:

$$A[a](t) = F\{ v(t) + B[a](t) \} \quad \text{for all actions } a \quad (7.1)$$

Where: $A[a](t)$ is the strength with which action a is selected at time t (contained in the real closed interval $[0,1]$);

$F\{-\}$ is a bounding function, simply restricting its argument to the interval $[0,1]$;

$v(t)$ is a random variable, usually normally distributed with mean 0;

$B[a](t)$ is the bias weight for action a at time t , an accumulated measure of the correlation observed between action a and reward changes (see below).

To correlate actions with subsequent evaluation changes, each element maintains a short-term memory, known as its eligibility, of the extent to which it has been active. When an evaluation change occurs, element biases are modified according to the extent of their eligibility. Mathematically the correlation bias weights are accumulated

as follows:

$$B[a](t) = B[a](t-1) + C[a] \{E(t)-E(t-1)\} TA[a](t) \quad (7.2)$$

for all actions a

Where: $B[a](t)$ is the bias toward action a at time t ;
 $C[a]$ is a learning rate parameter for the bias
weight of action a ;
 $E(t)$ is the feedback evaluation or reinforcement;
 $TA[a](t)$ is the eligibility of action a , an
exponentially decreasing weighted trace of values
of $A[a]$ before time t ; in the simplest case
 $TA[a](t)$ is merely $A[a](t-1)$.

At the start of the simulation experiment, the bias for each action is zero, favoring neither right nor left actions. During the exploration phase, the simulated experimental subjects move back and forth randomly between the lower large red and green regions of the environment (Figure 7.4) without reinforcement of any sort. Since reinforcement does not occur, nothing is predictive of reinforcement, and reinforcement is never predicted by the internal model component. Without reinforcement or its prediction, the action evaluation is always zero, and there can be no correlations between action and changes in evaluation during the exploration phase. Consequently, the bias weights remain zero. Once the testing phase has been reached, an internal model will have been constructed such that a correlation will exist through the internal model even in the absence of any external stimulation. This will result in the action selector converging on a preference for

one of the actions (this will be discussed in more detail later).

This sort of trial and error learning system is well known from the work on Harth's neural receptive field mapping technique ALOPEX (Harth, 1976; Harth and Tzanakou, 1974; Tzanakou, Michalak, and Harth, in preparation) and from the learning automata literature inspired by Tsetlin's work (Tsetlin, 1973). In a less simplified system than the network described here, it would be highly desirable to modify this action selecting mechanism so that it is able to use context information in selecting actions. This would allow it to learn to perform different actions in different situations or contexts without starting its search all over again each time the situation changed. Instead, it could remember for each context what actions were most successful in previous experiences. Trial and error learning mechanisms can be made sensitive to context in a fairly straightforward manner (see Michie and Chambers, 1968; Mendel and McLaren, 1970; and Section 5). However, it is not clear whether the actual current input, or the predicted input, or some combination of the two should be used as the context for the action selector. This problem with the current design is closely related to several others that emerge when sequences of actions need to be internally simulated in order to evaluate possible next actions. An

additional mechanism, such as a method for clearly separating actual from anticipated situations, is probably necessary to handle these cases. This example system is only a first step towards an adaptive network capable of creating and searching general internal models, and we do not consider these possibilities further. The artificial intelligence literature on planning would be highly relevant to future extensions of this adaptive network mechanism.

7.3.2 The Internal Model Component

The construction of the model of the world is a system identification task, and the solution adopted here follows

Kohonen's suggestion (Kohonen, 1977) for doing system identification using an associative memory. Kohonen's general idea was to train the associative memory with sample input to the system to be identified as the recall key, and to use the resultant output of the unknown system

as the training pattern to be recalled (Figure 7.7). If the unknown system has no memory (that is, it simply implements a function from input to output), then the associative memory will form a best least squares linear

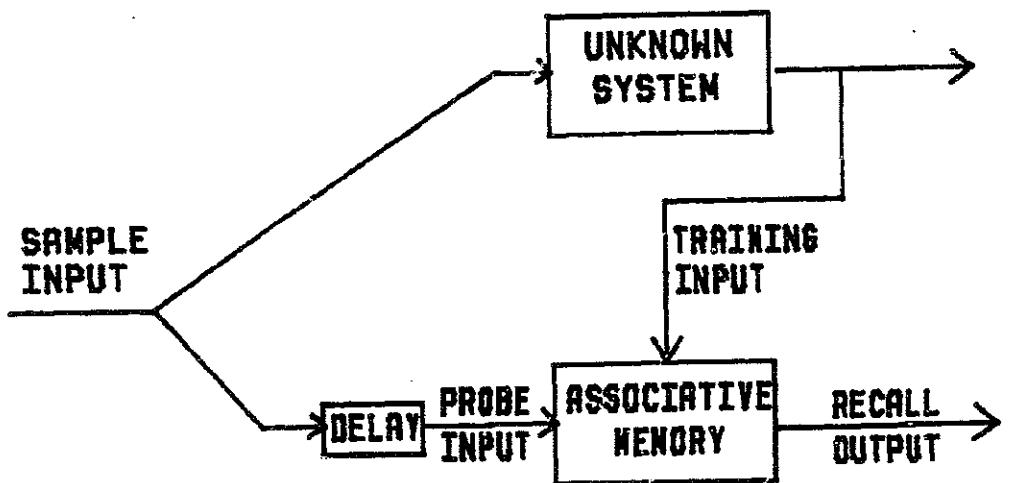


Figure 7.7. Kohonen's (1977) suggestion for doing input-output system's identification with a standard learning associative memory. The associative memory is trained by presenting paired samples of the input and output of the unknown system.

approximation of the unknown function. See Sections 2.4.1 and 4.5.

If the unknown system is the environment for an adaptive system, then this process will yield nearly the appropriate sort of model. Figure 7.8 is a slightly more detailed block diagram of the associative memory based machinery used in the simulated adaptive network for model construction and use. The associative memory in use here differs from the standard associative memories in being predictive: It produces as its recollection a prediction of what the next key will be. (In this sense it is similar to some of the early models of temporal associative memories; for example, Longuet-Higgins, 1968 a, b; Longuet-Higgins et. al., 1970.) Sections 4.3 and 4.5 more fully describe prediction.

Figure 7.9 shows a detailed wiring diagram of the model construction and readout machinery. This component consists of a bank of elements, each responsible for the prediction of a certain feature of the environmental stimulation, available in this case as the separate input lines for red, green, and reward stimulation. As a basis for making these predictions, each element is provided with the current action selection from the action selector component and the most recent predictions of stimulation.

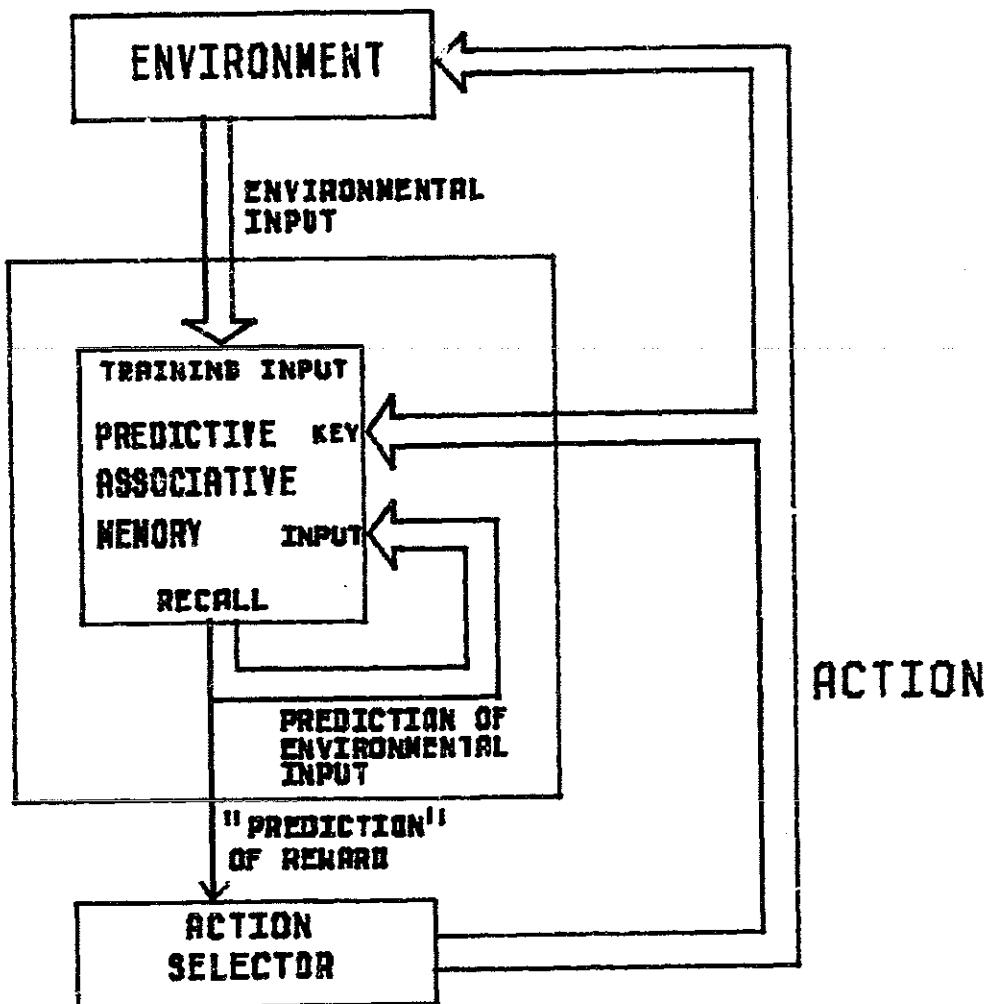


Figure 7.8. A more detailed block diagram of the adaptive network (cf. Figure 7.5) showing the central role of a predictive associative memory.

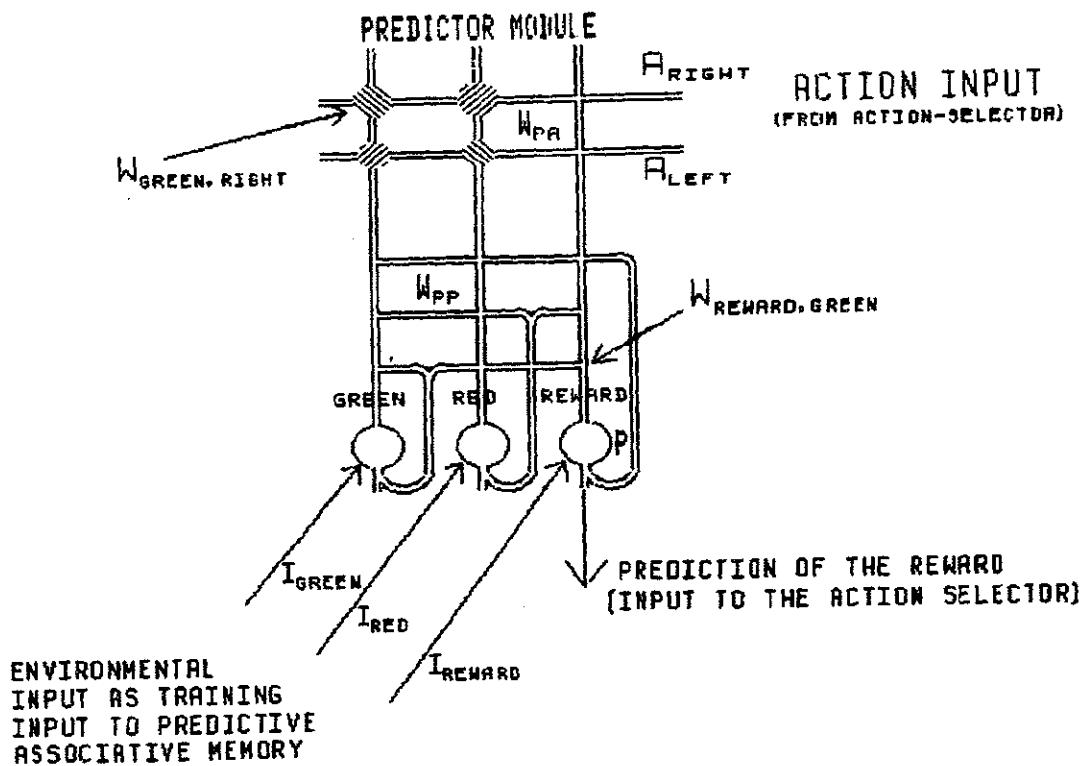


Figure 7.9. A more detailed wiring diagram of the model construction and readout mechanism. This component consists of a bank of elements, each responsible for the prediction of a certain feature of the environmental stimulation, available in this case as the separate input lines for red, green, and reward stimulation. As a basis for making these predictions, each element is provided with the current action selection from the action selector component and the most recent predictions of stimulation from the other predictor elements in this component. The fact that predictions of stimulation are used to make further predictions results in the recurrent architecture that we see in this network component.

from the other predictor elements in this component. The fact that predictions of stimulation are used to make further predictions results in the recurrent architecture that we see in this network.

These adaptive elements signal their predictions by responding as they would if the predicted stimulation were already present. For example, if the input information indicates that an element is going to receive strong excitatory stimulation, then the element becomes highly active immediately. This property, combined with the recurrent network architecture, results in the ability to chain predictive associations of as great a depth as there are features to predict (e.g., A predicts B, B predicts C, etc.).

The predictor elements used in this system are those extensively discussed in Section 4. However, although sufficient for the simple example system presented here, these elements may not be ideal for the purposes of constructing and using an internal model. Without going any further into the details of this element or the alternatives, we can note that these elements require the recurrent connections from each to itself to be made ineffective (zero and non-plastic) for effective operation in the architecture chosen for this network. This is easily

arranged, as it is in the simulated example system, but it is not an elegant solution, suggesting that there may be other hidden difficulties. This problem may suggest directions to proceed in deriving elements better suited to this purpose.

7.4 The Exploration Phase

During the random wanderings of each simulated adaptive creature in the exploration phase, the internal model component is forming a model to predict the red and green stimulation changes it experiences when it occasionally wanders into or out of one of the two colored regions. Figure 7.10 shows the state of the net near the end of the exploration phase for one of the simulated experimental subjects. The four relatively large connections from the actions to the green and red predictors of the predictor module indicate that the net has learned that right-moving actions predict increases in red stimulation and decreases in green stimulation, while left-moving actions predict increases in green stimulation and decreases in red stimulation.

The particular snapshot of the network activity in

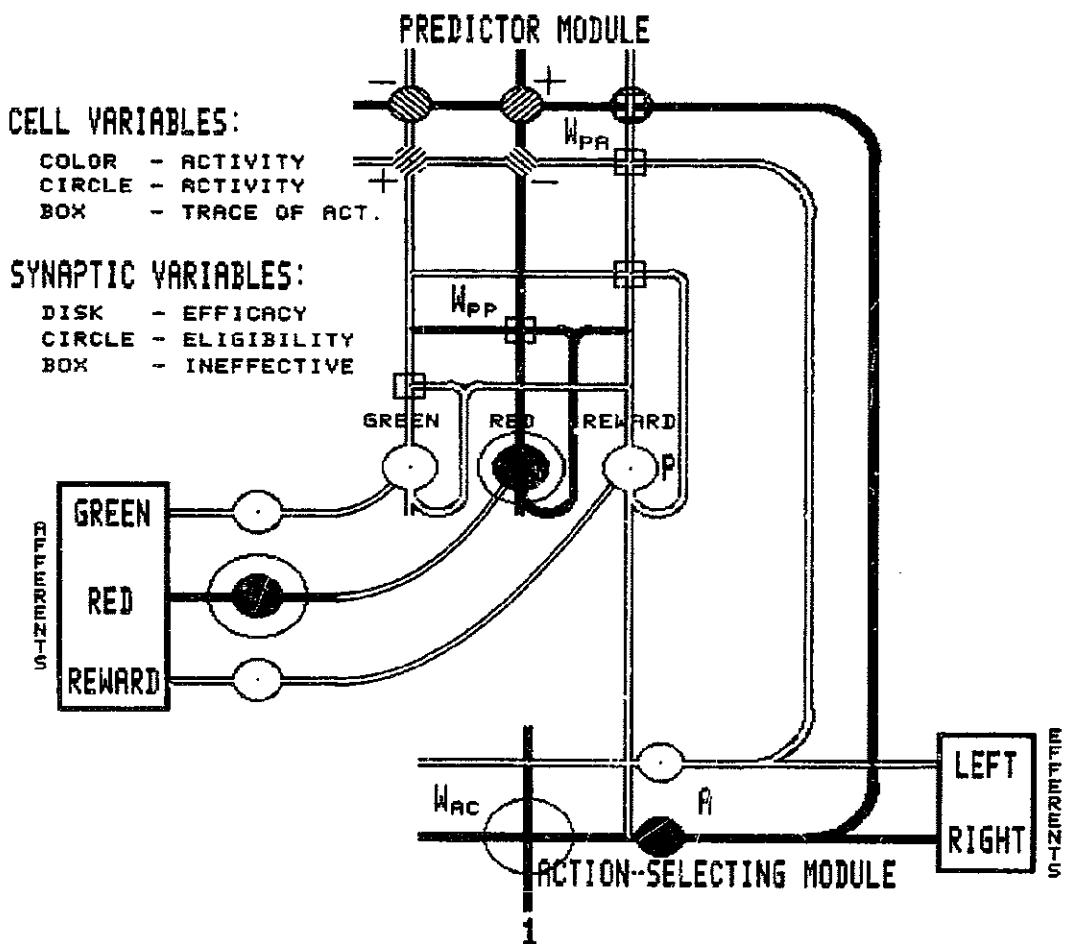


Figure 7.10. The state of the network near the end of the exploration phase for one of the simulated experimental subjects. The four relatively large connections from the actions to the green and red predictors of the predictor module indicate that the net has learned that right-moving actions predict increases in red stimulation and decreases in green stimulation, while left-moving actions predict increases in green stimulation and decreases in red stimulation.

Figure 7.10 shows how this knowledge is accumulated. The right-moving action has been selected more than the left-moving action in the last few time steps, as indicated by the greater eligibility of the connections from this action to the predictor elements, and in fact the subject has moved right during the most recent time step. This rightward movement has just brought the subject into the red region (this is indicated by the high activity in the red input while its trace of activity - indicated by the size of the box - is still zero). The resultant sensory input stimulates the red predictor element, causing an increase in its activity (indicated by the circle of this element being larger than the square), and this causes an increase in the eligible connections. The most eligible connections, as we have already seen, are those from the right-moving action. The net result is to further strengthen the pattern of learned associations that we allready see present in these connections.

7.5 The Association Phase

After 1000 time steps of the exploration phase, each subject is moved to the red goal box, left there for two time steps, and then provided with full reward stimulation.

Figure 7.11 shows the network state just after the reward is provided. Note the large positive connection from the red predictor to the reward predictor. The net has concluded that a prediction, or actual occurrence, of red predicts reward, since the red predictor element was highly active just prior to the increase in reward stimulation. The eligibilities of the connections from the red predictor are indicated by the large circles at these connections. Next, each subject is moved to the green goal box, left there for two time steps, and then the reward stimulation is removed. By a completely analogous process, the green prediction becomes a predictor of loss of reward, and the corresponding connection becomes negative (Figure 7.12).

7.6 The Representation Problem

This example system was constructed to be the simplest possible complete system capable of constructing and using an internal model. As such a minimal example, it only begins to address some of the critical issues involved. The simulated network was provided with a representation of the environment specially tailored to the task it was to solve. It had unique input lines for red and green stimulation, and the environment consisted only of areas that were entirely

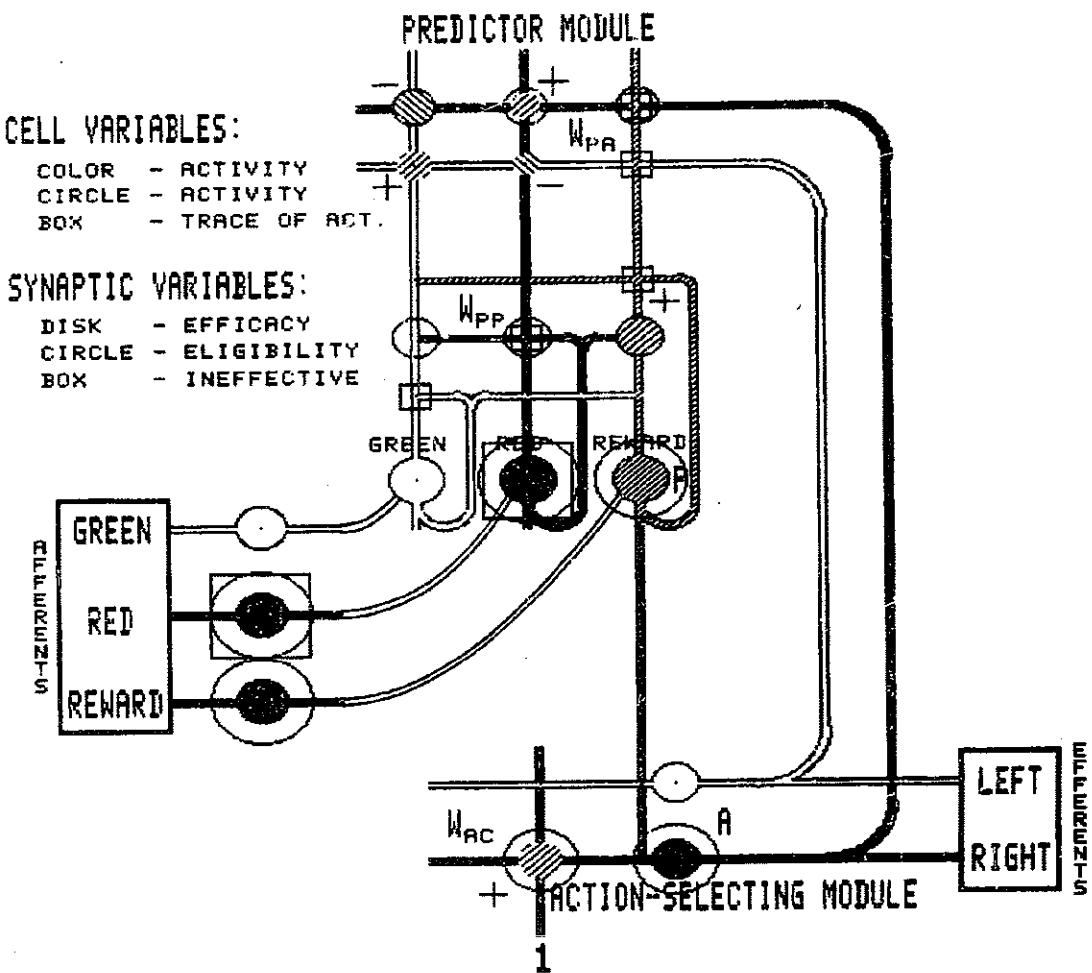


Figure 7.11. The network state just after the reward is provided. Note the large positive connection from the red predictor to the reward predictor. The net has concluded that a prediction, or actual occurrence, of red predicts reward, since the red predictor element was highly active just prior to the increase in reward stimulation. The eligibilities of the connections from the red predictor are indicated by the large circles at these connections.

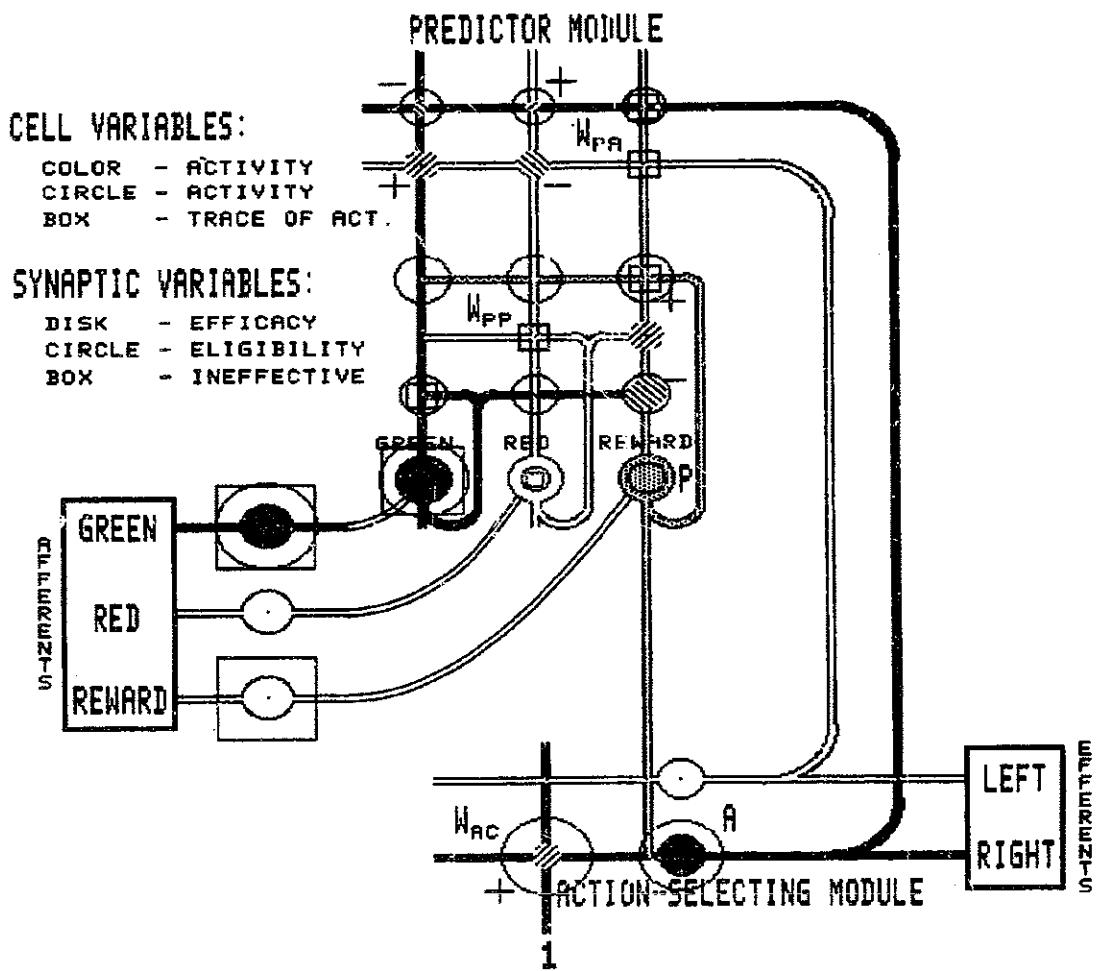


Figure 7.12. The network state after the reward has been removed while the subject was in the green box. Prediction of green has become a predictor of loss of reward, as indicated by the negative connection between the predictor elements for green and reward.

green, or entirely red, or neither. The relationships to be learned between actions and resultant stimulation, and between stimulations, were very simple ones in terms of the available action and stimulation representations. Mathematically, a network such as the one used here can only learn linear relationships between its representation of action, stimulation, and subsequent stimulation. To the extent that the actual relationships depart from linearity, such a network would be unable to form an accurate model.

One strategy for solving this difficulty is to retain the linear learning rules but to attempt to continuously evolve a representation compatible with that linearity. In general this is a difficult unsolved problem. Input features, output commands, and internal representations of environmental state (in the example system, environmental state was not necessary in forming a predictive model) all need to be developed. Probably the representation development is best done continuously and simultaneously with environmental interaction and the use of the model. This problem is closely related to the representation problem of artificial intelligence. Unfortunately, however, most of the AI work on the problem is unhelpful in that it merely attempts to find a good representation for a particular task rather than to find techniques for evolving representations in a more general setting. Genuinely

relevant work includes the feature extraction work in pattern recognition (Uhr and Vossler, 1961; Bledsoe and Browning, 1959; Klopf and Gose, 1969), Samuel's checker player (Samuel, 1959), Selfridge's pandemonium (Selfridge, 1959), and the work on non-linear associative memories (Poggio, 1975). A fundamental heuristic central to much of this representation development work is to direct the search for better representations according to which representation elements have already proved most useful.

Although the example network was given a sufficient representation ab initio, and has no capabilities for representation development, it does serve as a basis for considering what simultaneous environmental interaction and representation development may involve. In particular, we assume that there must be some property of the environmental interaction that indicates when and in what way the current representation needs to be changed. If this sort of example allows us to observe these properties in a simple case, then we will have made good progress toward making an adaptive network appropriately sensitive to them.

7.7 The Testing Phase

In the testing phase, each subject is returned to the initial location between the lower large red and green regions. As soon as the subject enters one of these regions the trial is over and the simulation is stopped. Of 200 subjects, 141 - over 70% - entered the red region first, a highly statistically significant result ($P < .005$). A second experiment was also performed in which the lower red and green regions and their barriers were removed during the training phase, but which was otherwise identical to the first experiment. The testing phase was halted after 300 time steps and the position of the subject was recorded. Each of the 100 subjects had moved far to the right, many times beyond the original location of the red region, at the end of that time. This indicates that the statistical nature of the primary result is due to random movements bringing some of the subjects within the green region, and thus ending the trial, before they have had enough experience with their internal models to be directed to the right.

7.8 Superstitious Learning

During the association phase reward is provided and then taken away from each subject while it is in the red and

green goal boxes respectively. During this time, whatever action the subject happened to select just before the reward stimulation is changed will be strongly reinforced or punished by that change. For example, the subject whose network state is shown in Figure 7.12 happened to associate the left action with reinforcement, as shown by the larger bias weight for the left action than for the right action. One might expect that the effect of these reward changes would be dependent on whatever action was randomly chosen just before the reward changes and that the effect on later behavior would thus be, on the average, symmetrical with respect to right and left moving actions. To ensure that this was the case, a third experiment was performed that was identical to the first one in all respects except that the action bias weights were set to zero just prior to the testing phase. This insured that there would be no initial bias either to the right or the left. Of the 100 subjects in this third experiment, just over 70% entered the red area first, confirming that the decision to move right can be made during the testing phase based completely on information stored in the internal model of the predictor module.

Figure 7.13 shows the state of the network of one of the subjects of the primary experiment several time steps into the testing phase but before either region has been

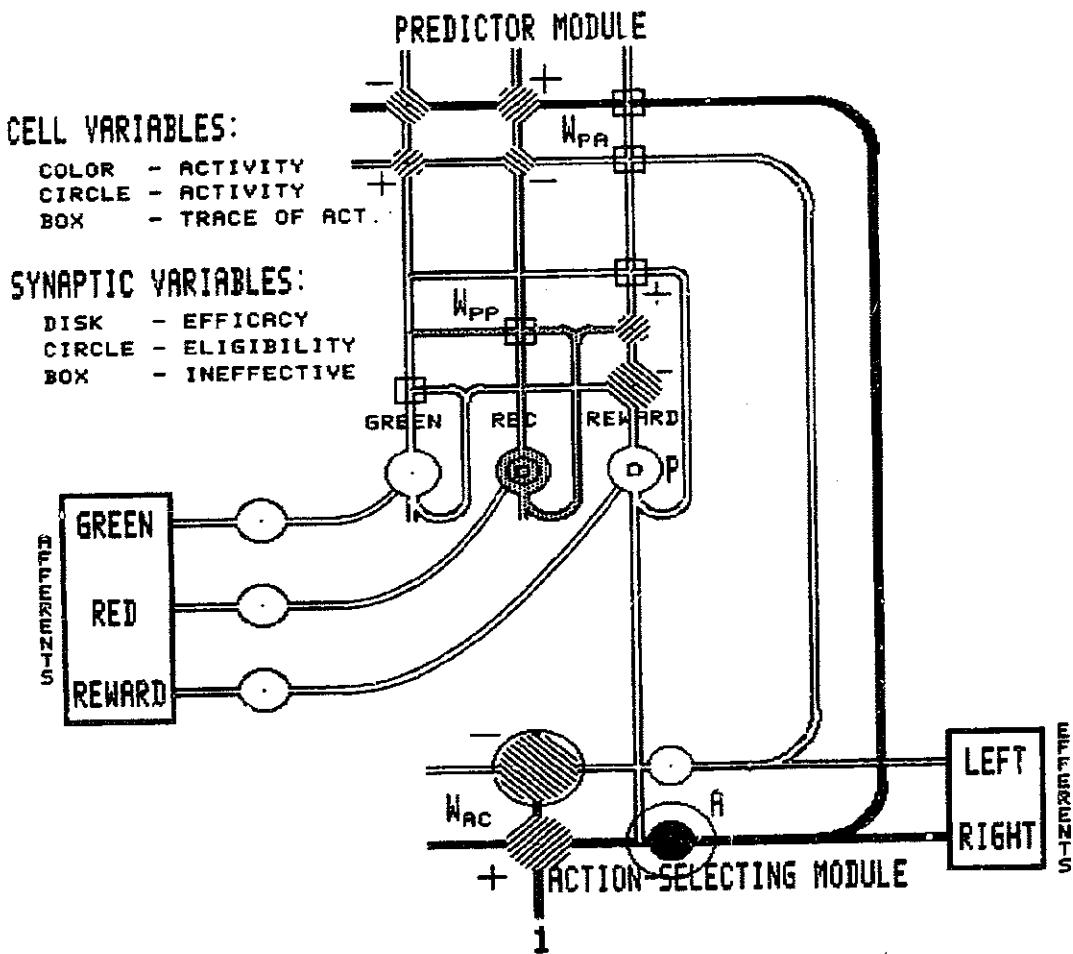


Figure 7.13. The state of the network of one of the subjects of the primary experiment several time steps into the testing phase but before either region has been entered. Notice that the bias weight for moving right (toward the red region) has become positive, whereas the bias weight for moving left (toward the green region) has become negative. The action selecting module happens to choose the instantaneous action causing movement right this time step. This selection results in an increase in the activity of the red predictor element, because moving right was found to be a predictor of red stimulation in the exploration phase.

entered. Notice that the bias weight for moving right (toward the red region) has become positive, whereas the bias weight for moving left (toward the green region) has become negative. The successive snapshots of network state in Figures 7.13 and 7.14 provide an example of how this comes about during the testing phase. In Figure 7.13 we see the action selecting module happening to choose the instantaneous action causing movement right. This selection results in an increase in the activity of the red predictor element, because moving right was found to be a predictor of red stimulation in the exploration phase. At the next time step (Figure 7.14), we see the prediction of red stimulation cycling around to activate the reward predicting element via the excitatory connection established during the association phase. Thus, as a consequence of the action selector's momentary choice of the right action, the predictor module, acting as an internal model, has generated the prediction of increased reward. Since the right action was selected at the previous time step, its bias weight is eligible (indicated by the large circle at its bias connection in Figure 7.14) when the prediction of increased reward arrives. Thus, this bias weight is increased, and the beast is further biased towards moving right. If on the other hand, the action selector had momentarily chosen the left action, green stimulation would have been predicted. This in turn would have predicted a decrease in reward.

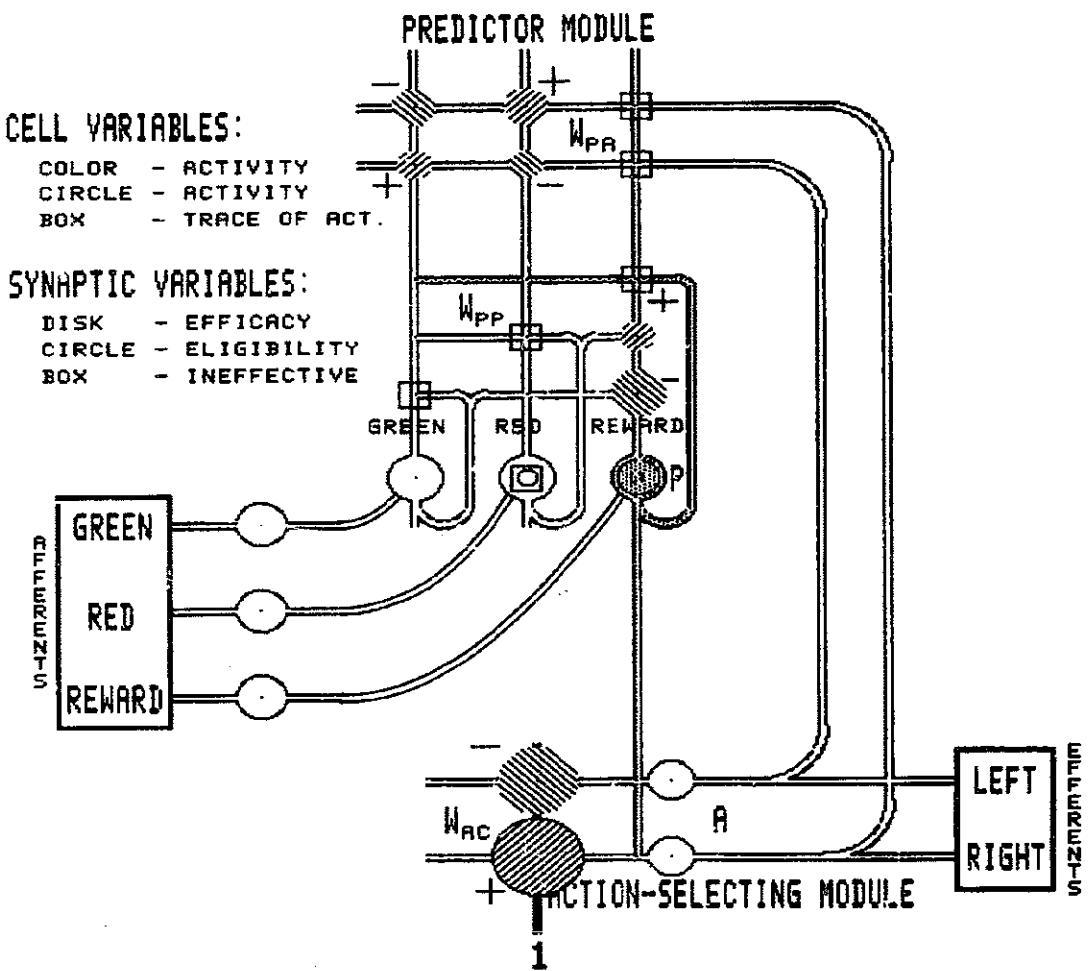


Figure 7.14. The state of the network one time step after Figure 7.13. The prediction of red stimulation cycles around to activate the reward predictor element via the excitatory connection established during the association phase. Since the right action was selected at the previous time step, its bias weight is eligible (indicated by the large circle at its bias connection in Figure 7.14) when the prediction of increased reward arrives. Thus, this bias weight is increased, and the beast is further biased towards moving right.

stimulation, and the selection of the left action would have been punished.

Recall that these momentary selections of right or left actions will not in general be accompanied by actual right or left movement. The environment responds to the actions selected in a relatively slow inertial manner: Several action selections in the same direction are usually necessary to cause actual motion in that direction. Both overt and covert actions are updated every time step of the simulation: The only difference is that the overt action, the "physical" movement, depends not only on the current covert action selection but also on past selections, weighted according to recency. Thus a consistently selected covert action becomes the overt action, while rapid fluxuations in covert action selection are averaged out. In this way the process of covert, internal trial and error via the predictive internal model can occur with relatively little overt action by the subject. No other delaying or decision making machinery is necessary to make the transition from covert thought trials to overt movement. In fact, the action selecting component (referring to Figure 7.5) is completely oblivious to whether it is receiving feedback from the external environment or from its internal model. From the action selecting component's point of view, the acquisition of an internal model merely means that the

feedback for its action selections returns more rapidly, significantly easing its problem of controlling that feedback.

7.9 Discussion

This section began with a presentation of philosophical and psychological views of thought as an internal modeling and simulation process. The construction of the example adaptive network presented here was guided by and exemplifies a theoretical perspective. In the authors' view, there are two novel aspects of this theoretical perspective: The nature and method of use of the internal model, and the way in which the construction and use of the internal model is coordinated with continuous environmental interaction. These two aspects are discussed further in the rest of this section.

This section has presented a method for adaptive control based on systems identification (model construction) that is extremely general (and apparently little investigated): 1) use repeated experiments with the input-output behavior of the system to be controlled to construct a model which yields similar behavior, and 2) to

select each control action, first interact with the model, in an input-output or black-box manner, to determine which action is optimal in terms of the model. The essential aspect of the model is that it is a behavioral model: Its successful use depends only on its input-output validity. The model can be interacted with to achieve an optimal response just as the external world is interacted with in the absence of a model. These perspectives on the nature and use of a model were summarized pictorially in Figure 7.1. It should be noted that appropriate general techniques for this sort of interaction with an environment or model are not currently well understood. This is an area of current active investigation by our research group.

Although we emphasize an input-output view of the internal model, this is not a return to the pre-state-space ways of thinking characteristic of the work of the 1950s. Such an internal model will in general include states (even though the model in the simulated example system did not). However, we do wish to emphasize that for this use of the model only the input-output aspects are important.

The second novel feature of this example network is the method used for coordinating interaction with the real environment and interaction with the model of that environment. Consider the approach taken in most artificial

intelligence (AI) systems for problem solving or reasoning about actions, such as SRI's famous robot SHAKY. SHAKY operates in three identifiable modes. In one, he visually scans the environment and constructs an internal model of it, aided by a priori knowledge and assumptions. In another mode, he uses his model of the environment and his current goals to perform a sophisticated search through the space of possible paths and actions. This search takes the form of an internal simulation with backtracking of many of the possible action paths. Finally, in the third mode, SHAKY shuts off his internal model and visual apparatus and executes "ballistically" his precomputed next action or series of actions. When the action is complete, or some unusual event occurs, SHAKY returns to the first mode. A lot of work in AI has concentrated on the model search step of the above scenario, without going any further towards coordinating the model interaction and the interaction with the real environment. By contrast, the example adaptive network presented here performs all three of functions - model acquisition, model interaction (search), and real time environmental interaction - simultaneously. If this example adaptive network is of interest, it is not because of its search capabilities, which are limited and primitive, but because it is a first step towards integrating the learning, search and use of internal models of the world. The fact that this integration was possible with little specialized

machinery - both adaptive elements used have been studied for more primitive purposes, and their interconnection pattern is not a highly restricted one - is a promising sign.

As mentioned earlier, the example network has been so constructed that the action selecting component (referring to Figure 7.5) is completely oblivious to whether it is interacting with the real world or the internal model. The effect of acquiring an internal model is merely that the thing with which the action selector is interacting begins to respond more rapidly to contemplated actions, and thus becomes easier to control. When an internal model is viewed from this perspective, it becomes clear that there is an even simpler case of the use of an internal model. A secondary reinforcer is an originally neutral event which has taken on reinforcing properties by virtue of being predictive of a primary reinforcer. To a network receiving this secondary as well as primary reinforcement, the development of the secondary reinforcer means that reinforcement for its actions arrives sooner following the actions than it did previously. The model consists of the rapid simulation of the tendency of the primary reinforcer to follow the secondary reinforcer. This results in an effective environment for internal action selecting elements that is more amenable to learning techniques. In this way

secondary reinforcement can be seen as a very simple case of the construction and use of an internal model.

SECTION 8

GOAL-SEEKING SYSTEMS OF GOAL-SEEKING COMPONENTS

8.1 Introduction

Klopff (1972, 1979, 1981) has put forward the hypothesis that neurons are goal-seeking components and that an understanding of neural function requires us to view animal brains as goal-seeking systems each of whose primitive components possesses its own local goal and adaptive machinery that makes progress toward that goal possible. In this section we discuss, in rather broad form, some of the conclusions reached in our study of how goal-seeking systems might be constructed from goal-seeking components. We address such questions as: How can components possessing their own local goals and means for approaching them interact so as to produce goal-directed behavior at a higher level? Can we expect to produce intelligent computer systems from goal-seeking components? Are there existing theories that are relevant to these issues? Has this approach already been tried and largely abandoned due to

lack of progress? Is there support for the hypothesis that animal intelligence arises from the interaction of goal-seeking components? More specifically, is there support for the idea that neurons are goal-seeking organisms? We cannot as yet answer all of these questions in unqualified terms, but as a result of our research we are able to frame some of them in forms that can provide a rich basis for future research. Our discussion of these issues will touch on many different areas, ranging from the mathematical theory of games to the biochemical regulatory mechanisms of single cells. It is hoped that this far ranging, and still largely speculative, essay will provide the reader, as our research has provided us, with a vivid sense of how intelligence could have evolved, how it might be understood, and, perhaps, how some of its more subtle characteristics might be produced artificially.

8.2 Goal-Seeking Components

In Section 2 we discussed in rather technical form various types of goal-seeking behavior. Only some of these forms, we will argue here, can be expected to yield interesting forms of higher level behavior in collections of goal-seeking components. The key issues involve the forms of environments in which a component can succeed in achieving, or in making progress toward, a goal (or goals),

and the kind of information a component can take advantage of in the goal-seeking process. Some types of goal-seeking systems can succeed only if their environments always act in very restricted ways. For example, a thermostat can achieve its "goal" of maintaining a room's temperature near a certain set-point only if its environment cooperates by always reacting to the thermostat's control signals in the manner anticipated by the thermostat's designer. Its success depends entirely on the adherence of the environment to restricted forms of behavior. Here we discuss a number of different types of environments, in increasing order of generality, that a goal-seeking component might face. We end the discussion with the observation that the environment confronted by a living organism is not likely to be of the restricted form required for the success of most previously studied goal-seeking strategies.

8.2.1 Learning with a Teacher

Many of the systems commonly studied as goal-seeking systems can only succeed if their environments always provide them with very "high quality" information about what actions they should take in various situations. Typically, adaptive systems of this kind produce a response to a stimulus pattern, and then are told by their environment how far and in what direction they must change their response in

order to be correct (a signed error signal). Some systems have two possible responses and are told whether their responses were right or wrong (e.g., the perceptron of Rosenblatt, 1962) [footnote]. The system's goal is achieved when the error is zero for its response to each stimulus pattern.

In order for an environment to provide reliable error signals, it must, in effect, "know" what each response ought to be. The stimulus patterns for which the environment knows the correct response, and can therefore provide an error signal, constitute a "training sequence" (cf. Nilsson, 1965). Success is measured solely by how closely the system's responses match the correct ones during the training sequence. The interest in a goal-seeking system of this type is its ability to generalize. After achieving its goal of zero error for the training sequence, its responses to stimulus patterns for which the correct responses are unknown by the environment provide possibly correct (and

It is important to note that being told if one is right or wrong in the case of only two alternative responses is equivalent to being given a signed error. In fact, it is equivalent to being told exactly how you should have responded. We discussed this fact and the misunderstanding of the perceptron learning rule arising from it in considerable detail in Section 2.

possibly incorrect!) generalizations of the environment's knowledge. But here is the rub. Many environments will not be able to provide a training sequence because they may not know any more than the system about what responses ought to be made.

This type of goal-seeking behavior is often called "learning with a teacher". Teachers interact with their students in many different ways, but what is meant here is that the teacher knows the answers to a set of questions and provides the student with very informative error signals that the student tries to reduce to zero. In Section 2 we pointed out that most such formal teacher-student interactions are equivalent to those in which the teacher simply provides a set of questions together with their correct answers (and the student computes its own error signal). We therefore arrive at this curious dilemma: These types of goal-seeking systems can only solve problems about which so much is known beforehand that explicit and detailed instructions can be given.

8.2.2 Learning without a Teacher

A digression is in order here to consider what is sometimes called "learning without a teacher". This term has been applied to the problem of classifying input

patterns in the absence of error signals or indications of correct responses. The object of this type of system is to form its own classification scheme based on some measure of similarity between input patterns. Patterns that are similar to one another are to be placed in the same class. This problem is also known as the problem of clustering. The term "learning without a teacher" applied to this problem is misleading in the context of our discussion. We are discussing goal-seeking systems that receive varying degrees of help from their environments, and a clustering algorithm would appear to require no help at all from its environment. This is true, but it is also true that its environment has nothing to do with the success or failure of the clustering system since no interaction with the environment is involved. It is an open-loop problem. While clustering may be an important part of a learning system, we are interested in problems in which a system's environment does determine success or failure but provides little explicit help to the system about how it should act.

8.2.3 Learning with a Critic

A type of goal-seeking system that can achieve its goal in less cooperative, or less knowledgeable, environments is one capable of "learning with a critic" (to use the terminology of Widrow, 1977). The environment of this type

of system need only evaluate the system's response to each sensory situation. The crucial point here is that an evaluation can be made by an environment that does not know what each correct, or best, action is. The critic may not know enough about the problem in order to say, in effect, "do this in response to this input." Without this knowledge, however, it may still be able to say, "whatever change in behavior you just made was an improvement." A system whose goal is to maximally satisfy an environmental critic must effectively search through its repertoire of actions for that which optimizes whatever evaluation function the critic happens to be using at any time. This type of goal-seeking behavior is sometimes called "reinforcement learning" (e.g., Mendel and McLaren, 1970). The critic can be viewed as providing "rewards" and "penalties" to the goal-seeking system when its behavior becomes, respectively, better or worse with respect to whatever criterion is being used for evaluating actions. Alternatively, one could effectively place the critic inside the goal-seeking system and speak of the system as possessing a preference ordering of its inputs. Using this formulation, we say that the goal of the system is to cause its environment to provide inputs that are maximal according to this order. This latter view is more general and was adopted in our more technical discussion of Section 2.

Reinforcement learning systems are capable of using the critic's information to provide knowledge beyond that possessed by the critic. They can determine the actual structure of good responses. A critic need not have this knowledge from the start, but a teacher, in the sense described above, must start with this knowledge for a rich set of cases.

In many cases the environment of a goal-seeking system may not even be able to provide evaluations of the system's actions that are constant and reliable. A goal-seeking strategy that can make progress under the guidance of a reliable critic may not be sufficient to operate with an unreliable, or noisy, critic. Problems characterized by constant but noisy critics are sometimes called "decision problems under uncertainty," and strategies that are useful under these circumstances are a step more general than those requiring consistent evaluations. Methods for succeeding in these types of environments are fairly well studied as in the theory of learning automata which commenced with the work of Tsetlin (1974). Another way in which an environment can be less helpful is by being able to provide evaluatory information only occasionally. Achieving the goal of satisfying an occasional critic presents very considerable difficulties which have not been surmounted by any general methods. It is this type of environment to which we now

turn.

8.2.4 Learning with an Occasional Critic

Although a system capable of learning with a critic need not have an environment as cooperative or as knowledgeable as that required for learning with a teacher, its environment must still provide helpful evaluatory information. What if the environment provides a critic, but one that only occasionally offers advice? Or, viewed in a slightly different manner, what if reward and penalty events occur only seldomly? A goal-seeking system that performs well under the guidance of a constant critic may not have much of a chance of finding action sequences leading to such isolated rewarding events. This involves the well-known "apportionment of credit" problem. If a rewarding event does occur (e.g., winning a game of chess), how do you apportion credit among all of the various actions taken before the reward? This is closely related to the problem of optimizing an evaluation function that has large "plateaus" or "mesas"; that is, one that shows no variation over large areas (Minsky and Selfridge, 1960). The search for peaks in an evaluation function is greatly facilitated by the presence of broad "foothills," and in their absence a search method can become inefficient to the point of being useless.

One form that a partial solution to the apportionment of credit problem can take is the effective "creation" of foothills surrounding the isolated peaks provided by an occasional or late critic. In order to do this a system must be sensitive to a multitude of environmental signals in addition to the critic's occasional evaluation. This information can form the substrate out of which "foothills" can be constructed. If a system can learn from its experiences that a particular type of action in a particular type of sensory situation can make the occurrence of a rewarding event more likely, then it can interpret the occurrence of that sensory situation itself as a step in the right direction. Of course, the environment must contain regularities that can be discovered and exploited, but these regularities need not be "prepackaged" and transmitted over predetermined teacher or critic channels. In other words, the environment of the component may contain many sources of information some of which can act as useful critics, but the component does not know from the beginning what sources of information are important in this way. In addition to learning what it must do to satisfy the occasional critic, it must learn to recognize other sources of useful guidance information.

8.2.5 Nature as an Occasional Critic

Most organisms find themselves in environments that act as occasional critics. The environment implicitly acts as a critic by occasionally providing sensory stimulation that acts, through genetically determined mechanisms, as reinforcement. (We can avoid the considerable controversy surrounding the meaning of reinforcement by accepting, for our present purposes, the operational view that a reinforcing event is any event that alters the likelihood of the organism's preceding actions. Dennett's, 1978, essay "Why the Law of Effect Won't Go Away" provides an excellent view of why some events might be reinforcing for animals.) But only rarely is an animal provided with a neat reinforcement gradient that can simply be climbed. This does happen in the animal kingdom, as when a bacterium encounters a nutrient gradient leading up to the most profitable place to eat, but the world is usually not so helpful. The world does provide, however, a wealth of information that can be used to "figure out" how to cause reinforcing events. Experience can show, for example, that if food is seen in the distance, then eating it can be made more likely by moving in its direction! An organism (of sufficient complexity) is able to construct for itself the advice a useful critic would provide by using the wealth of information its senses, tuned by its evolutionary history,

provide. We think that one of the primary uses of associative learning capabilities is to permit a goal-seeking system to, in effect, create its own foothills out of originally neutral sensory stimulation.

This is the sort of goal-seeking behavior that our interpretation of Klopf's hypotheses attributes to individual neurons. They are organisms that seek reinforcement and are able to obtain it in environments that do not provide explicit help. Early adaptive network theorists did not study networks of components having this kind of robustness.

8.3 Networks of Goal-Seeking Components

We cannot characterize any precise level of goal-seeking capabilities that components must possess in order for an interacting collection of them to exhibit "higher-level" goal-seeking behavior, by which we mean goal-seeking behavior that is more sophisticated than that of which the components themselves are capable. Instead, we see a spectrum of possibilities in which the potential for interacting collections to exhibit emergent behavior increases as the sophistication of the components increases. We have a sense of some extreme cases. A computer, for example, is constructed from non-goal-seeking components

and, although it can exhibit a very high level of behavioral complexity, cannot be said to be a goal-seeking system (although it can certainly be programmed to become one). A network of components each of which requires an explicit and knowledgeable teacher in its environment (e.g., a network of perceptrons) tends not to produce behavior of a significantly higher level than that exhibited by the components themselves. At the other end of the spectrum, we see that human societies are capable of solving problems that individual humans cannot, and, if Klopf's neural hypothesis is correct, then the brains of the more phylogenetically advanced animals provide other examples of collections of goal-seeking systems that can solve more complex problems than their components can solve.

We think that the characteristics of component systems most relevant to their potential for interacting to exhibit higher-level behavior are as follows:

- 1) How much knowledge a component's environment must have, or how much cooperation it must give, in order for the component to be able to make progress toward its goal.
- 2) What amount of information the component can take advantage of in pursuing its goal.

We think that as components become more capable of adapting in less structured environments and come to use more environmental information in pursuing their goal, the

likelihood that they can form the basis of networks with higher-level behavior increases. We think it is the conjunction of these two properties that is important. A high level of performance in one of these areas without a high level in the other (if this situation is even possible) would, we think, be insufficient.

We are not able to prove this conjecture, but we can indicate why we think it is true. A component of a complex system has as its environment aspects of the whole system's external environment and an environment provided by its interactions with other components. If each component always acts in its own self-interest, then the environment of any given component cannot always be explicitly helpful. In fact, due to the self-interest of all of the components, any component's immediate world may be a rather hostile place. A component must not be completely stymied by this adversity but must do the best it can under the circumstances. But how can a group of selfish individuals possibly form a coherent structure? The answer lies in the possibility for cooperation. In some circumstances the components of a cooperative group can achieve more progress toward their own goals than they would be able to achieve by always unilaterally acting for their own best interests. It is a high degree of inter-component communication that forms the substrate in which cooperation can take place. Much of

the rest of this section deals with what is meant by cooperation, how it can come about, and how it can lead to systems capable of solving problems that the individual components, working alone, cannot solve.

8.4 Games and Cooperation

If we are to study how goal-seeking behavior can arise from a collection of goal-seeking components, each operating solely according to its self-interest, we must develop a view of what might constitute the goal of the collection. In our discussion of goal-seeking components, we assumed that the progress of a component toward its goal could be determined by the values of an error-signal (in the case requiring an explicit teacher) or a payoff, reinforcement, desirability criterion, or performance index (in the case of learning with a critic, or reinforcement learning). The goal of a component is to minimize or maximize (as the case may be) this criterion. In most formal studies the desirability criterion is modeled as a number, or at least is measured on an ordinal scale, so that it is clear what constitutes an improvement in performance. But when can we say that the performance of a collection of components, each having its own desirability criterion, improves?

The generalization of the problem of optimizing a

single desirability criterion to the case of many desirability criteria constitutes the Theory of Games of von Neumann and Morgenstern (1943), also known as Multicriterion Decision Theory, and, extended to dynamical cases, as Generalized Control Theory (Ho, 1970). It is game theory that is most relevant to the study of goal-seeking systems of goal-seeking components. In fact, it requires a formulation as general as a game to even begin to express with any precision notions such as coalition, team, cooperation, trust, and threat. These concepts become meaningful the moment one considers optimization under more than one criterion of performance. This situation arises in organizations of all types, including societies, and indeed any collection of goal-seeking components. If neurons are in fact goal-seeking organisms, then the concepts of game theory also provide a starting point for the study of neural organization (and it would not be surprising that social analogies to neural function seem appropriate as in Crane, 1978).

Game theory has never had a significant role in studies of machine intelligence. We think this is largely due to the fact that game theory provides help in formulating problems but provides very little help in actually solving them. For example, game theory says almost nothing that is useful for the design of a chess playing program. But when

we consider the construction of problem solving systems from components that are robust adaptive systems, then, by the technical definition of a game, such a system plays a game, and the adaptive strategies of the components collectively provide an algorithm for playing the game. Games that arise in this way can be far simpler than games like chess or checkers, but successful strategies for playing them can provide solutions to what are far from trivial problems. Game theory "applied" at this level is beginning to find important applications in the field of problem solving by distributed processing systems (e.g., the approach to access control of Yemini and Kleinrock, 1978, and Brooks, 1980) and in the study of evolution (e.g., Dawkins, 1976; Maynard-Smith, 1978). We believe that concepts from game theory are relevant for the study of brain function, but they require some notion of local goals in order to be applicable in their technical sense.

8.4.1 Group Optimality

Consider a collection of goal-seeking components in which each component receives its own payoff or reinforcement signal from its environment, or, more generally, in which each component has its own preference ordering on its inputs. Further, suppose that the input a component receives depends not only on its own actions but

also on the actions of the other components. This situation is a game. A component's payoff is a function of the actions of possibly all the other components. Each component is a "player" in the game. If the preference orderings are identical for each member of the collection and each receives the same input from its environment, then optimality for the group is achieved by maximizing this common preference ordering. In this case of components with no conflicts of interest the problem reduces to the single criterion case we have already considered. Such a group is often called a team (Marshak and Radner, 1972). The associative search network that we described in Section 5 is a team of adaptive elements in this sense.

If the preference orderings of the components differ, the situation becomes much more complicated. The input situation that is best for one component may not be best for another. In other words, it may be impossible for the collection to act so as to maximize all of the preference criteria at the same time. An example is provided by the special case of a zero-sum, two-person game in which what is best for one player is, by definition, worst for the other. Other examples occur whenever the preference orderings depend on the allocation of limited resources (e.g., resources in an economic system, computer processing resources, transmission time in a packet radio network). In

fact, multicriterion decision problems that involve conflicts of interest, or require tradeoffs, are probably the rule rather than the exception in most domains.

We want to describe two game theoretic concepts that provide a starting point for considering how a system of goal-seeking components might itself be goal-seeking. The first is the notion of an equilibrium point (sometimes called a Nash point) of a game. An action of the system (consisting of the individual actions of each of its components) is an equilibrium action if no single component can improve its own local payoff by unilaterally changing its own action. Examples of this kind of equilibrium may be provided by the driving styles that predominate in various cities. The aggressive driving style of many large cities might be considered an equilibrium point of some appropriately defined game of transportation. Any individual driver takes serious risks by driving less aggressively than most other drivers. More efficient transportation may be possible if all, or most, drivers use different styles, but in the absence of this kind of collective decision, it is optimal for a driver to drive aggressively (because everyone else does).

For many games there exist collective actions that are more preferable to each of the players than any of the

equilibrium collective actions. In other words, a collective action that is stable with respect to unilateral individual actions (an equilibrium) does not necessarily provide the highest payoff to each of the players that is possible. For example, traffic flow might be improved so that nearly every driver can expect to get to his destination more quickly if nearly every driver adopted a driving style more courteous than the aggressive equilibrium one. But a player can take advantage of this kind of situation by acting so as to improve his own performance at the expense of other players (a few aggressive drivers among many cautious ones can often make very good time).

A collective action that a single individual can improve upon according to its own preference ordering only at the expense of others is called pareto-optimal. Some pareto-optimal actions in a game may be better for all players than any of the possible equilibrium actions (those that are better constitute the game's negotiation set), but these actions are unstable. It requires cooperation among the players in order for these collective actions to be maintained. In a game the term cooperation is applied to any means of introducing dependencies among the actions of the players (and is therefore used in a sense more technical than the common usage). These interdependencies may be enforced by pre-play communication leading to a binding

agreement on a collective action. All forms of cooperation require inter-player communication.

Precisely which collective action constitutes the best and most equitable solution of a general n-person game is not automatically clear. Various proposals have been made, but each has certain shortcomings (see Luce and Raiffa, 1957). It is clear, however, that the solution should belong to the game's negotiation set and that cooperation among the players is required to achieve it in the general case. Unless the goal-seeking components of a system cooperate, they must generally settle for lower individual payoffs than are possible.

8.4.2 Goal-Seeking Systems as Game Players

One of the fundamental assumptions of all the classical game theoretic studies has been that the players know the entire structure of the game from the start. That is, they know what payoff each player will receive for each possible combination of individual strategies. The emphasis of these studies is largely on the explication of what constitutes the game's solution and all of this concept's associated complexities. There has been relatively little discussion of algorithms for attaining a solution (whatever it may be) for the case in which the structure of the game is not known

a priori. In this case, the players must accumulate knowledge about the game by performing a collective action, receiving their respective payoffs, incorporating the knowledge so gained into their decision algorithm, and then playing again. The problem of finding a game's solution via "iterative-play" under these restrictions is the multicriterion generalization of the usual function optimization problem, and it is this problem in which we are most interested. The general paradigm suggested here is recognizable as the "generate-and-test" procedure that forms the basis of many Artificial Intelligence programs.

The research begun by Tsetlin (1974), concerning what have since become known as "learning automata," illustrates how goal-seeking components that are sensitive only to their own payoff signal (and no other input information) can function as players in a game. Narendra and Thathachar (1974) provide a good review of this research. A collection of goal-seeking components, each capable of improving its performance in a sufficiently general type of environment (in the case of learning automata, environments that provide a constant but unreliable critic), will naturally converge to an equilibrium play of the game. A collection of such independently operating goal-seeking components is not able, however, to converge to any solution that is not an equilibrium point since they do not communicate with one

another. We do not know of attempts to extend this research to produce systems that are capable of finding a game's better-than-equilibrium solutions. Some means of inter-component communication must be introduced.

We think that the adaptive elements that we have developed based on Klopf's notion of a heterostat have precisely the characteristics that will permit collections of them to achieve solutions in a game's negotiation set. These adaptive elements are sensitive to more information than just their own payoff signals. This context information can include information about what actions other elements are performing.

8.5 Coalitions and Cell Assemblies

For the "iterative-play" or "generate-and-test" paradigm which we have been discussing it is possible to give a more concrete view of forms that cooperation can take. One way to improve a generate-and-test strategy is to find some way of being very selective in generating structures. One should try to test only structures that have a high likelihood of being improvements over structures already tested. One of the most frequently suggested methods of doing this is to generate new structures that are novel recombinations of parts of structures that have

already proven successful. Additional power can be achieved by somehow identifying what parts are likely to be important. Selfridge (1978) describes this approach, and it has been studied extensively by Holland in the study of analogs of the process of evolution through reproduction and natural selection (Holland, 1975).

A striking and easily observed example of the recombination approach is seen in the evolution of television programs. Television programming has become an almost explicit example of a single criterion optimization problem: Maximize the Nielson rating. Since a television series is characterized by many attributes, this optimization problem can be viewed as a special case of a game in which each player (whose actions are the possible values of one of the attributes) shares the same payoff measure or preference ordering. (Actually a television network is attempting to optimize its entire schedule, and an explicit game is played with the other networks. In addition, other criteria, such as production cost are also involved.)

It is increasingly clear how network programmers generate new series. One strategy is to produce a program that exaggerates certain features of an already existing and successful program. One sees sequences of programs

containing more and more of certain characteristics. This is straightforward hill-climbing and does not necessarily involve cooperation among program attributes. Another method is to produce programs that are novel combinations of attributes of programs that have already proven successful (e.g., "Police Woman"). A very successful program will generate numerous "spinoffs" which have its major attributes mixed with a variety of others.

A recurring group of attribute values characterizing a successful program and frequently occurring in numerous spinoffs might be viewed, from a game theoretic point of view, as the action of a cooperating collection of players - a coalition or, in genetics, a co-adapted set of alleles. From an individual player's point of view, it is better to perform a particular action when the other players in the group are performing certain specific actions. If players' actions are attributes for television programs, then this cooperative process will cause constellations of features to occur together frequently. These constellations can then be used as parts of the recombination process which can, in turn, recombine to form higher level parts. This leads to a generate phase of the generate-and-test paradigm that tends to produce structures having high likelihoods of success.

This provides a rough view of one way in which

cooperation in the game theoretic sense can occur and what problem solving advantages it may provide. There is a close relationship between this view and cooperativity in neural systems. If the goal-seeking players are neurons, one arrives at a view of cell assemblies as coalitions in the literal game theoretic sense. The view has been put forward, notably by Arbib (1978, 1981a, 1981b) and Amari and Arbib (1977), that important phases of neural processing might usefully be thought of as types of relaxation processes in which consistent interpretations of input mutually excite one another and compete with rival interpretations through inhibitory interactions. The reticular formation model of Kilmer, McCulloch, and Blum (1969) is the first neural model to exhibit this form of cooperation. Studies based on this view provide interesting examples of how cooperation can arise in plausible neural architectures. They do not, however, make contact with the literal game theory notion of cooperation since the goal-seeking nature of the components is not made explicit. We think that these studies provide an important part of a more general view of neural cooperation in which there is a sense of "why" components might come to interact in this manner.

8.6 Heterostats as Cooperating Game Players

We mentioned above that studies of goal-seeking components as game players show that, when acting in parallel with no inter-component communication, they are able to converge to one of a game's equilibrium points. Tsetlin's learning automata studies, for example, show this (Tsetlin, 1974). Not one of these studies, however, considers the case in which the goal-seeking components are able to communicate among themselves. Tsetlin mentioned the potential importance of inter-component communication but did not pursue it.

The adaptive elements that we have developed based on Klopff's heterostat concept are sensitive not only to reward/penalty signals but also to other signals that provide information about the sensory situations in which actions are performed. These adaptive elements are capable of learning to perform the optimal action in each sensory situation (under certain conditions) as illustrated by the associative search network described in Section 5. In the study reported there, the adaptive elements comprising the network were not interconnected, and we assumed that the sensory situation in which an element acted was provided solely by an input vector generated external to the network. However, if recurrent connections were to exist among the

adaptive elements, then the input to each adaptive element could depend on the internal situation consisting of the actions of the other players as well as the external environment situation. These recurrent connections can form the basis of cooperative behavior.

Suppose, for example, that a group of adaptive elements fire together at a particular time, and that this activity pattern (i.e., the network's collective action) produces a response from the environment that ranks high according to the preference orderings of the elements in the group. Since each element fires in the context of the others' firing, the high preference measure will cause excitatory connections to form among the elements in this group (according to the learning rules we have been using). The result is a cell assembly that will tend to become active if any of its constituents become active. Similarly, if the firing of a group of elements results in a response of low preference, then the elements will tend to become mutually inhibitory.

The development of cell assemblies in this manner is similar to the theories often proposed, beginning most explicitly with Hebb (1949). However, what we are suggesting takes an important step toward making the intuition behind these theories more precise. If the

elements are goal-seeking systems, we can say that a cell assembly forms because coordinated activity furthers progress toward the goals of the elements forming the assembly. The use of closed-loop learning rules (Hebb's suggested learning rule is open-loop) allows us to make this view explicit. In order to make the notion of cell-assembly-as-coalition more than a superficial metaphor, it seems necessary to endow the components with their own local goals. Further, if the elements are capable of achieving progress toward these goals in a general class of environments, and can communicate with one another, then one would expect such elements to assemble because it is better for them if they do.

8.7 Cooperation by the Creation of Environments

The formation of cell assemblies, or coalitions, as a form of cooperation is perhaps the simplest form that cooperative behavior might take. It requires each component to sense the actions of other components in order to detect situations in which its activity will yield a high payoff. Goal-seeking components can interact in another way if they are able to provide each other not only with neutral context information but also with rewards and penalties. Klopf has suggested that components ought to be able to communicate with one another by means of signals that can take on

reinforcing qualities, a capability he described as the ability to use generalized reinforcement (Klopf, 1972, 1981). We indicated above that this capability can provide a means for a component to construct for itself the advice a useful critic would provide if a constant or reliable critic were not available in its environment. But the ability of components to communicate via non-neutral signals can provide the basis for other forms of organizational capabilities.

Suppose we want a goal-seeking system to do something for us. If we know what the system's goal is, can sense the system's actions, and have enough control over the system's inputs, we can arrange contingencies in its environment to cause it to do something that we want done as it pursues its own goal. By doing this we, in effect, use its goal-seeking capabilities for our own ends. We create an environment for the system in which its goal and our own are the same. Since a goal-seeking system may have capabilities that we do not have ourselves (it may have access to information we do not have and have specialized control capabilities), we are able to cause problems to be solved that we are not able to solve ourselves. We can "tap" another goal-seeking system's goal-seeking capabilities.

8.8 Control Strategies for Problem Solving

The preceding discussion related cooperativity in games to the generate-and-test paradigm which has been exploited in many artificial systems. We viewed the generate-and-test procedure as a means for finding structures that optimize some measure of desirability or performance. By restricting ourselves to this paradigm, we did not wish to imply that it is the only paradigm in which the problem solving power of a system of goal-seeking components can be manifested. While we do think that the generate-and-test paradigm is basic to problem solving procedures (Dennett, 1978, argues that it is the only way to create novel solutions), it provides only one part of efficient problem solving control strategies. The "monolithic" application of generate-and-test to a problem (formulated as an optimization problem as discussed above) is likely to disregard important information that can be used to guide the solution procedure.

All that we have said about generate-and-test, however, can be extended to the problem of adaptively forming more complex strategies for problem solving if we recall that the adaptive power of the components we have considered permits them to use information other than just a measure of performance. They are able to learn to perform the optimal action in a variety of input situations as illustrated by

the associative search network (Section 5). This additional information can indicate, along with other things, the state of the problem solving procedure. In other words, the search performed need not be a "blind" search as suggested by the stark form of the generate-and-test paradigm.

An example is useful here. Imagine the problem of tuning a television, all of whose controls are very far from their optimal settings. This task requires a search through the multidimensional space of control settings for the setting that maximizes some measure of picture clarity. It is usual to formulate this task as a function optimization problem. But this formulation applies equally well to the problem faced if we had access to the controls and were provided with a meter that registered "picture clarity" but were not allowed to see the picture itself. In actuality, the problem we solve is really quite different. We do see the picture, and we make extensive use of information it provides other than just its clarity. We quickly gain a sense of what effect each knob has on the picture. If the picture is rolling, we can (after sufficient experience) directly alter the appropriate control. Similarly, if it is too dark, we can directly change that too. The picture provides information that tells us what to do. Different actions are appropriate for different pictures; that is, in different sensory situations. Consequently, tuning a

television is not a standard function optimization task but is instead an example of what we have called an associative search task (Section 5). In solving it, we use a strategy more complex than a unitary hill-climbing procedure. (In fact, labels such as VER and BRT on the control are useful only because this is the case: These words do not describe a scalar clarity measure but rather describe the picture itself.) The strategy is to search for the best rules of the form "in situation X, do Y" for a given set of situations.

It is our impression that there have been two approaches to solving these kinds of problems and that they have not been integrated as thoroughly as they could be. One approach, as we have just discussed, is to view these problems as function optimization tasks in which much of the available information is neglected. The other approach focuses on control strategies for applying a given set of rules, or "productions," of the form "in situation X, do Y." In this approach the function optimization aspects of the problem tend to be ignored since either a fixed set of productions is used or only simple methods are used for generating new productions. We think that a combination of these two approaches can yield generally useful methods for solving problems of this kind. Our associative search network (Sections 5 and 6) illustrates some of the possibilities in a simple form.

8.9 Neurons as Goal-Seeking Systems

Klopf hypothesized that neurons are goal-seeking systems that are able to make progress toward their goals in a rather general class of environments and that possess sensitivity to wide ranges of contextual information. Klopf has argued for the biological reality of this hypothesis by indicating the kinds of data it might make understandable (Klopf, 1972, 1981). In Section 4.7 we put forward hypotheses about how neural mechanisms could implement adaptive strategies of the required complexity.

However, all of this empirical support is, at best, circumstantial: Certainly many other hypotheses are consistent with this range of observations. Here we discuss another line of support which, while certainly indirect and speculative, seems to us to be particularly compelling. This is an evolutionary argument based on the adaptive capabilities observed in freely living unicellular (or acellular) organisms. Neurons are not, of course, freely living organisms, but it seems plausible to us that they possess mechanisms that are not too distantly related to those of unicellular organisms.

In their classic work The Orientation of Animals: Kineses, Taxes and Compass Reactions (1961), Fraenkel and

Gunn discussed a number of methods used by animals for finding and remaining near light or dark areas, warm or cool areas, or, in general, for approaching attractants and avoiding repellants. One of the most primitive mechanisms is a strategy that they called klino-kinesis. The most intensely studied example of klino-kinesis occurs in the behavior of various types of bacteria such as Escherichia coli, Salmonella typhimurium, or Bacillus subtilis. This manifestation of klino-kinesis, known as bacterial chemotaxis, was discovered in the 1880's and was recently reviewed by Koshland (1979). These bacteria propel themselves along relatively straight paths by rotating (!) a flagellum. With what at first appears to be random frequency, they reverse flagellar rotation, thus causing a momentary disorganization of flagellar filaments. This causes the organism to stop almost instantaneously and tumble in place. As the disorganized flagellum continues to rotate in the new direction, its filaments reorganize causing the organism to be again propelled along a straight path. Consequently, flagellar reversal causes a random change in direction of travel.

Adaptively useful behavior results because the frequency of flagellar reversal is modulated by the direction of movement with respect to levels of attractants and repellants. Reversal frequency decreases if movement is

toward higher attractant concentrations and increases if movement is toward lower concentrations. Repellants have a similar effect, mutatis mutandis. This modulation of flagellar reversal biases locomotion so that the organism approaches and remains near places of maximal attractant concentration or minimal repellent concentration. It is a very effective strategy, particularly when gradient information is very noisy. Koshland (1979) describes this type of behavior and the underlying biochemical mechanisms in great detail. Selfridge (1978) emphasizes the generality of this type of adaptive strategy, which he calls the run and twiddle strategy, by describing it as follows: If things are getting better, keep doing what you are doing; if things are getting worse, do something else.

One sees in these single cells the existence of goal-seeking behavior. The receptor repertoire and chemotactic responses of various species of bacteria indicate that they either move toward chemicals that are needed for survival or, more generally, move toward conditions that favor their survival (Koshland, 1979). It is completely clear, moreover, that the strategies used to make progress toward these goals are closed-loop strategies that require short-term memory in order to detect gradients. Changes in a cell's receptor activity are caused by the cell's actions by means of a chain of influences that passes from the motor

apparatus, through the external environment, and then back to the input apparatus of the cell. Memory is required because this feedback path requires time to be completed. The locomotory manifestation of this adaptive strategy in a freely swimming organism makes the closed-loop nature of the interaction obvious. Moreover, it is just this sense of what a bacterium is doing that renders the regulatory mechanism intelligible. It would be much more difficult to understand this mechanism if one had to consider it completely outside of its role in guiding locomotion. There would be no sense of its function and adaptive significance. Neurons, on the other hand, continue to be studied without consideration of the possibility that important information is passing from the neuron, through its environment, and then back as modifications in afferent signals.

Chemotactic responses have been suggested as possible mechanisms for guiding fiber outgrowth during neural development. Although numerous trophic factors may be involved, no conclusive experimental support for this hypothesis seems to exist (see Lund, 1978). We are not suggesting, however, that neurons necessarily use literal forms of chemotactic responses to guide growth and migration during development. Rather, we are suggesting that in fully developed nervous systems neurons may use closed-loop adaptive strategies similar in logical structure to

chemotactic strategies. Instead of actual spatial movement there need only be a kind of logical or virtual movement as a neuron's output influences its input. Like bacteria, neurons possess receptors located in their membrane, or just inside, that detect chemical signals from their environments. The sensory processing system produces signals that control the motor response of the bacterium by altering the probability of flagellar reversal. Neurons similarly respond to afferent signals, transmitted by chemical means, by means of chemically mediated processes whose details are not yet understood, and produce "motor" responses consisting of action potentials. Of particular interest is the fact that some bacteria respond to changes in membrane potential in the same way they do to changes in attractant or repellent levels. In B. subtilis, for example, increases in membrane potential cause tumble suppression ("running") and decreases cause tumble generation ("twiddling") (Miller and Koshland, 1977) (cf. Klopf's hypothesis about neural goal-seeking behavior.) What tends to be disregarded in the study of single unit information processing is the possibility that important aspects of a neuron's behavior involve its ability to influence its own input when operating in its usual environment.

We think that the similarity between the mechanisms

producing goal-seeking behavior in freely living cells and machinery within the neuron provides the most promising line of support for the hypothesis that neurons do implement closed-loop, goal-seeking strategies. We think, along with Koshland, that the continued study of the numerous commonalities between bacterial chemotaxis, and other simple forms of adaptive behavior in single cells, and the signaling systems of neurons is a promising avenue for future investigation. We hope that our theoretical research will help lay the groundwork for thorough empirical investigation.

8.10 A Sense of Neural Function

Maintaining the evolutionary point of view hinted at in the preceding section, we can see the outlines of a vivid sense of neural function. Let us suppose that as neurons became specialized in fast electrical signaling, they did not lose all of the properties of their less specialized ancestors. Let us suppose that these ancestors were able to follow chemical gradients in their fluid environments, approaching some chemicals and avoiding others, using hill-climbing strategies implemented by mechanisms not unlike those we see in present day bacteria and other unicellular organisms. Since these strategies obviously confer great adaptive advantages, we would expect them to be

refined and extended by the evolutionary process. We therefore arrive at the view that neurons are using strategies closely related to those that are successful in promoting the survival of freely living cells. The environments in which neurons "swim," however, consist of the very complex and abstract contingencies of the brain of which they are a part and, more indirectly, of the organism and its environment to whose survival they contribute. Of course, we do not mean "swim" in a literal sense, but the term provides a vivid image of the essential closed-loop nature of a neuron's interaction with its environment. The consequences of motor output are felt, at varying later times, as changes in the patterns and intensity of afferent activity. When a cellular action is followed by cellular sensory reception indicating an increase in the concentration of a particular chemical, then we can think of that action as causing a kind of virtual movement up a virtual concentration gradient. Mechanisms that cause real movement toward attractants and away from repellants in real spatial concentration distributions can do the same in these virtual distributions if the closed-loop dynamics are similar to those produced by movement in space. Some neurotransmitters may act as "attractants" and others as "repellants" for the neuron in its virtual spatial environment. This would mean, simply, that the neuron would act so as to increase its stimulation by some transmitters

and decrease its stimulation by others.

We could imagine a further step in abstraction in which the attracting and repelling qualities of certain chemicals were transferred to the electrical events that tended to co-occur with chemical reception. This might have provided important increases in processing speed. Among the functions neurons perform is that of relaying chemical signals at high speeds (diffusion need only occur across synaptic clefts). In this role, a neuron acts as a kind of high speed "repeater" of chemical signals. Electrical signals could therefore be viewed as representations of chemical signals that can be transmitted more quickly and manipulated more easily than the chemical signals themselves. Analogs of chemotactic mechanisms could then provide control mechanisms for "swimming" in virtual concentration distributions represented by electrical potentials. The control of the feedback dynamics of a neuron's environment by other, perhaps phylogenetically later, neural levels could provide a means for tapping primitive adaptive capabilities to provide parts of the solutions to complex problems. We are imagining a situation in which a higher-level center might pose a problem to a lower center in the form of an environment having particular dynamical characteristics. By "swimming" in this environment guided by its own goal-seeking strategies, each

cell in the lower-level center contributes to a cooperative solution of the given problem.

Based on existing experimental data, this is an admittedly speculative view of neural function. But we also do not know of any strong experimental counter evidence. While the importance of closed-loop control processes is very well recognized, both at a basic biochemical level and at the behavioral level as in the study of insect optomotor responses, it is our impression that the possible closed-loop nature of a neuron's interaction with its environment is not a familiar concept. Experimental paradigms designed to study single units tend to break the feedback pathway through a neuron's natural environment. Our observations from computer experimentation with artificial closed-loop adaptive strategies indicate that systems which appear quite simple when embedded in appropriate feedback can appear much more complex when observed in open-loop mode. This suggests that in order to understand the information processing capabilities and adaptive mechanisms of neurons, it may be necessary to gain a sense of what kinds of environments form their natural habitats. This is not so problematic in the case of freely living unicellular organisms since the dynamics of their spatial environments are similar to those of our own. It is much more difficult to understand the intricate contingencies of a neuron's world.

CHAPTER 9 CONCLUSIONS

The major objective of this project was to assess the promise of constructing adaptive systems from adaptive components based on Klopf's (1972, 1979, 1981) theory of heterostatic components. It quickly became apparent that a great variety of issues were involved. A methodology evolved in which we attempted to isolate the various features of Klopf's hypothesis and study each of them in as stark and as simple a form as possible. We attempted to determine exactly which behavioral capabilities of a variety of learning rules were due to which specific features. As a result, we have experimented with adaptive elements that differ from Klopf's hypothesis in numerous ways. As is perhaps common for this kind of analytical methodology, we have not given equal time to the process of reassembling our findings into a simple and unified picture. Nevertheless, we can state some overall conclusions.

9.1 What is New?

Some of the intuition underlying the notion of constructing goal-seeking systems from goal-seeking components has always played a role in studies of "self-organizing" systems. However, early studies are characterized by the use of components whose capabilities are too limited to support network behavior beyond a rather low level of sophistication. In the most general terms, these components required from their environments a great deal of explicit help in order to make progress toward their goals. In order to provide this help, a component's environment must know more about the problem's solution than is generally possible (Section 8.2). The central idea of the research reported here, on the other hand, is that the components of an adaptive system must be robust enough to be able to make progress toward their goals in environments that are unhelpful, indifferent, or even hostile.

Our research has given us a strong impression that adaptive network research was left in a very primitive state when emphasis shifted to the more symbolic approach that characterizes most current Artificial Intelligence research. This is not so much a criticism of these earlier studies; they were necessary beginnings. Rather, it leads us to question the tendency to dismiss the entire network approach

based on the lack of dramatic success in the first attempts. In the years since these attempts, considerable sophistication has been achieved in the field of computer science. It now seems clear that the problems to which early adaptive network efforts were directed are too difficult to be so quickly solved by any approach. We cannot claim to have solved these general problems by the research reported here, but we can claim to have shown that some important features were absent from earlier network studies.

We have uncovered several widespread misconceptions about the nature of adaptation and learning. These misconceptions are largely due to an overestimation of the generality of particular learning rules or of particular theories. We think that the pervasiveness of the following fallacies has greatly hindered progress:

- a) The perceptron learning rule and similar stochastic approximation methods solve problems that are open-loop problems, or can be recast as open-loop problems without additional assumptions. These rules are not adequate models of animal learning behavior in instrumental conditioning experiments, but are more closely related to classical conditioning (Section 2.4.4).
- b) The formulation of adaptation as function optimization

is too abstract to shed much light on most forms of learning. It does not permit the importance of information other than payoff or reinforcement information to be considered. This type of "blind search" is not always necessary in applications (Sections 2.4.5 and 8.8).

c) Two very different types of search problems are usually confounded. One type, which we called error-correction, is characterized by the fact that the desired situation can be recognized as such when it is first encountered. These problems can therefore be solved without evaluating all possible situations. Negative feedback techniques, whether explicit or in the guise of gradient descent procedures, are associated with these types of searches. An extremum search problem, on the other hand, is not fully solved until the entire range of possibilities has been explored (although in practice such full solutions are not generally possible). In the former type of search, optimality is a local property of individual trials, whereas in the latter type, it is a property of the entire set of possible trials. This rather subtle distinction is perhaps most important in distinguishing the research reported here from other adaptive network research (Section 2.3.1.4).

d) The view that adaptation can be equated with equilibrium-seeking (as in "homeostasis") is misleading. Equilibrium-seeking involves error-correction search rather than extremum search. It is an important but restricted

process (Sections 2.3.1.4 and 2.5).

Much of the criticism of the approach to developing intelligent systems based on numerical, data-directed methods typified by the perceptron rests on the difficulty in extending these methods to solve more difficult examples of the same types of problems they were already solving. For example, implications of the perceptron's limitation to forming linear discriminant functions were pointed out by Minsky and Papert (1969), and the shortcoming of hill-climbing methods for the optimization of functions with large plateaus or many false optima were pointed out by Minsky and Selfridge (1960). The criticisms we have implicitly made in this report are of a completely different kind. We have pointed out the restricted nature of the problems these methods were designed to solve rather than their limited ability to solve them. We, of course, agree that general pattern recognition and function optimization problems are very difficult to solve completely, but we think problems of this difficulty need never occur. Pattern recognition is usually just one part of a complex adaptation or learning task, and the function optimization task is so abstract that the formulation of a problem as such a task usually requires potentially valuable structure and information to be ignored. It seems to us that sophisticated adaptive behavior can result from a system

designed to solve a variety of interrelated adaptation and learning tasks, each of which is relatively simple. In other words, when formulated in an appropriate manner, sophisticated adaptative behavior need not require any single subsystem to form highly nonlinear discriminant functions or optimize functions having broad plateaus or many extrema.

It remains for future research to provide substantive support for this claim. The ability of an adaptive system to learn to exhibit overall nonlinear behavior clearly remains necessary, and the research reported here does not demonstrate how this is possible. We do, however, believe that the groundwork has been done by our exploration of novel types of adaptive elements. Linearity and nonlinearity are not properties of problems or control tasks per se but are properties of particular representations of them. The search capabilities of the adaptive elements studied here suggest that representations can be adaptively formed in which the necessary discriminations can be simply made.

The approach to adaptive network design suggested by Klopff appears to be novel for the following reasons:

- a) The components suggested by Klopf are most closely related to the control theoretic notion of reinforcement learning control systems (Section 2.4.10). These systems combine pattern recognition, function optimization, and control functions so as to solve control problems about which there is little a priori knowledge. It is novel to consider adaptive networks composed of components as sophisticated in their capabilities as even simple learning control systems (such as the various heterostat formulations with which we have experimented).
- b) The type of component suggested by Klopf combines the capabilities of components previously studied. Components such as the perceptron classify input vectors but do not conduct extremum searches; that is, they are not reinforcement learning systems. Components such as the learning automata of Tsetlin and his school (Tsetlin, 1973) perform extremum search but are not sensitive to information other than the reward/penalty signal and therefore do not perform pattern discrimination. The components on which our research has focussed combine these capabilities.

9.2 Open-Loop Learning

At an early stage of our research, we devised a learning rule having several interesting properties even though it went only part way toward including the features

required for genuine closed-loop reinforcement learning. This is the learning rule described in Chapter 4 where we related its behavior to animal behavior in classical conditioning experiments. From the perspective of the entire research effort, this learning rule is of interest largely for the following reasons:

- a) It permitted us to make strong contact with animal learning data and the Rescorla-Wagner model of classical conditioning.
- b) It permitted us to gain an understanding of some of the consequences of Klopf's notion of eligibility in a context relatively free from the complication of other issues.
- c) We concluded that by basing adaptive changes on the deviation of reinforcement level from an average of past levels (or the "expected" level) together with eligibility, one obtained a stable, well-behaved rule that also produced a variety of interesting effects (notably predictive behavior and stimulus context effects).
- d) The predictive behavior of these adaptive elements suggests that they can be used to represent knowledge of environmental contingencies in a form permitting its access for real-time decision making. If the predicted consequences of a particular action taken in a particular situation were available before the actual consequences,

then decisions could be made by evaluating proposed actions before they were executed. This type of internal model use has often been discussed, but we were able to show how a simple network could implement it (Chapter 7). Although our demonstration of this capability remains in extremely simple form, we think that the principles illustrated can be extended.

9.3 Generalized Reinforcement

Klopf distinguished his heterostat component from others previously studied by emphasizing its property of generalized reinforcement as contrasted with restricted reinforcement. A restricted reinforcement component has specialized positive and/or negative reinforcement inputs in addition to excitatory and inhibitory inputs. There is a consequent sharp distinction between the "teacher" as the source of reinforcement signals and other types of information. A generalized reinforcement component, on the other hand, has the property that all (or many) input signals are potential reinforcers. In the course of our research it became apparent that Klopf's proposal contained novel features even without considering the property of generalized reinforcement. We recognized that an extremum seeking component that is also sensitive to information other than a reinforcement signal had unexplored

implications even with reinforcement arriving over a specialized channel. Adaptive elements previously studied (e.g., the perceptron) have specialized "teacher" channels, but the signals arriving over them are error signals rather than reinforcement signals. Consequently, the networks described in Chapters 5 and 6 (associative search networks) consist of components with specialized reinforcement channels. These simulations illustrate novel capabilities without the additional complication of generalized reinforcement. We regarded an understanding of these capabilities to be a logical prerequisite to tackling the more general case. We have not yet determined what additional adaptive power generalized reinforcement may provide.

We have, however, gained a fairly clear view of what issues generalized reinforcement involves. These are most clearly discussed in Sections 3.4.5 and 8.2.4 (learning with an occasional critic). In environments in which pure or "wired-in" reinforcing events occur only occasionally, simple hill-climbing strategies are not effective in causing the reinforcing events to occur. We see the role of generalized reinforcement as the construction of the advice that a constant and reliable critic would provide if such a critic were available as an initially identifiable source of information. The system need not know "who the critic is"

from the beginning. Indeed, a critic may not exist as a prepackaged source of information. It is well known that the ability to form its own performance evaluation function is a very important feature of a sophisticated adaptive system. Samuel's famous checker playing program, for example, most strongly relies on this type of learning (Samuel, 1959).

We think that the following ideas are likely to play roles in the elucidation of these issues: a) Prediction and the use of predictions of reward as rewarding events themselves; b) The notion of secondary reinforcement from animal learning theory; and c) The problem of "mesas" in function optimization. Our discussion of prediction and higher order learning in Chapter 4 is most relevant to these issues. It is a point of interest that our development of components which combine pattern discrimination and extremum search sets the stage for a concrete investigation of these additional features, but only a few steps have been taken in the research reported here.

9.4 Associative Search

The network which we called the associative search network (Chapter 5) is the result of placing components capable of both pattern discrimination and extremum search

in a paradigm that has become well known among present day adaptive network theorists. We were thus able to clearly demonstrate what additional capabilities these components provide. Research on associative memory networks has generated a relatively recent and sizable literature. These networks have aroused interest because they can successfully retrieve information under noisy conditions and are insensitive to various degrees of localized damage. We have not discussed these capabilities at length since good treatments are available elsewhere (e.g., Anderson, et al., 1977; Kohonen, 1977; Palm, 1980). The associative search network retains all of these features but has the additional ability to determine for itself what information should be stored by conducting searches through the set of possible associations and retaining the most highly rewarding ones. Thus, in addition to questions about how information is stored, questions about what information should be chosen for storage are addressed. This permits applications of associative memory systems to a wider class of problems than previously possible. The landmark learning problem (Chapter 6) provides a simple illustration.

9.5 Biological Implications

Although progress toward an understanding of the cellular basis of animal learning is proceeding at a rapid

rate, it is still premature to propose detailed hypotheses about the biological mechanisms that might implement the learning procedures we have studied. We have speculated about what mechanisms could be involved (Section 4.7), but the possibilities still remain too numerous to warrant the singling out of any specific, detailed neural model. Nevertheless, our theoretical study leads us to make several observations. First, the general failure to find clear examples of associative learning at the synaptic level may be due to the possibility that learning is taking place there that is more complex than simple association rather than less complex. If associative learning requires short-term memory at a cellular level (as it would if the adaptive elements we have studied were implemented at a cellular level), then one would not expect to observe it experimentally unless the cell's internal state and the context of the stimulation could be controlled. Second, it may be profitable to conduct experiments designed to test cellular responses in closed-loop situations. Closed-loop studies of unicellular organisms have led to an understanding of adaptive mechanisms that would have remained obscure if the organisms were always observed outside of their natural closed-loop relationships with their environments (Sections 2.4.8.1 and 8.9). Neurons may use similar adaptive strategies to control their environments.

The importance of closed-loop control processes in biology is well recognized. At the most basic level, the notion of feedback regulation in chemical reaction systems is a central concept in biochemistry. At the behavioral level, it is very clear that certain forms of behavior are explicitly directed toward controlling input. For example, insect optomotor responses reveal their function clearly when their influence on the environment is permitted to be reflected as changes in sensory input. Despite the ubiquity of control concepts in biology, it is our impression that the possible closed-loop nature of a neuron's interaction with its environment is not a familiar concept.

Experimental paradigms designed for the study of single neurons tend to break any feedback pathway through a neuron's environment. It is often useful to break the feedback loop of a control system in order to experimentally determine the details of its control law. Breaking the loop permits the experimenter to exert complete control over the system's input (this, for example, is what voltage clamping accomplishes). But these open-loop studies of feedback control systems are generally useful only after it is realized that the functionality, and perhaps the adaptive significance, of the system manifests itself only when the control loop is in place. Otherwise, the open-loop observations are likely to appear complex and confusing.

This is especially true for closed-loop systems that are adaptive or are capable of learning. This suggests that in order to understand the adaptive properties of neurons, it may be profitable to design experiments which permit a neuron's output to influence its input.

9.6 In Summary

As a consequence of our research, we believe that considerable adaptive power can be achieved by systems composed of goal-seeking components, provided the components possess sufficiently robust adaptive capabilities. Previous adaptive network studies have considered components having only limited adaptive power. We have shown that components designed with attention to the temporal dimensions of information processing can behave as simple reinforcement learning control systems. These components acquire knowledge about feedback pathways in which they are embedded and use this knowledge to seek preferred inputs. Simple networks composed of these components can solve types of problems that are completely beyond the capabilities of networks studied in the past. Although we believe these results to be novel, they represent only a small step. Much remains to be done in furthering what we believe to be a promising approach to distributed, adaptive systems.

APPENDIX A

ANALYSIS OF STEADY STATE BEHAVIOR OF THE RESCORLA-WAGNER/WIDROW-HOFF PREDICTOR FOR A SIMPLE CASE

For simplicity we treat the continuous time case in which a trial consists of a single impulsive CS of amplitude α at time $t = 0$ and a single impulsive UCS of amplitude λ at time $t = T$. Letting the time functions x and z respectively denote the CS and UCS signals, then for $t \geq 0$:

$$x(t) = \alpha \delta_0(t) = \begin{cases} \alpha & \text{for } t = 0 \\ 0 & \text{otherwise,} \end{cases}$$
$$\text{and } z(t) = \lambda \delta_T(t) = \begin{cases} \lambda & \text{for } t = T \\ 0 & \text{otherwise.} \end{cases}$$

Let the element's output be the linear result of one CS input pathway:

$$y(t) = \begin{cases} \alpha w(0) & \text{for } t = 0 \\ 0 & \text{otherwise.} \end{cases}$$

For continuous time, the Rescorla-Wagner/Widrow-Hoff Predictor rule (Equation 4.12) becomes:

$$\frac{dw}{dt} = c[z - \bar{y}] \bar{x} \quad (\text{A1})$$

where \bar{x} and \bar{y} are respectively the eligibility generated by x and the expectation generated by y . We assume that \bar{x} and \bar{y} are exponential traces of their respective variables.

That is, let

$$\bar{x}(t) = \alpha e^{-\gamma t}$$

$$\bar{y}(t) = \alpha w(0)e^{-\xi t}$$

where γ and ξ are positive decay rates. Then Equation A1 becomes

$$\frac{dw(t)}{dt} = c[\lambda \delta_T(t) - \alpha w(0)e^{-\xi t}] \alpha e^{-\gamma t} \quad (\text{A2})$$

Here we investigate the conditions under which a trial leaves the associative strength of the CS unchanged; that is, we ask what initial weight $w(0)$ is such that $w(t) = w(0)$ for some time t occurring after the trial. But when is a trial over? Weight changes can occur as long as \bar{x} and \bar{y} are not both equal to zero and thus can occur during the ISI and after the UCS offset ($t = T$). Since exponential traces never return to zero, we consider the case of an infinite intertrial interval and ask what $w(0)$ should be so that

$$\lim_{t \rightarrow \infty} w(t) = w(0)$$

Integrating Equation A2 we obtain:

$$\begin{aligned} (\lim_{t \rightarrow \infty} w(t)) - w(0) &= \int_0^{\infty} \frac{dw(t)}{dt} dt \\ &= c \int_0^{\infty} [\lambda \delta_T(t) - \alpha w(0) e^{-\xi t}] \alpha e^{-\gamma t} dt \\ &= c \lambda \alpha e^{-\gamma T} - c \alpha^2 w(0) \left(\int_0^{\infty} e^{-(\gamma + \xi)t} dt \right) \\ &= c \lambda \alpha e^{-\gamma T} - \frac{c \alpha^2 w(0)}{\gamma + \xi} \end{aligned}$$

Then $w(0)$ must be such that

$$c \lambda \alpha e^{-\gamma T} = \frac{c \alpha^2 w(0)}{\gamma + \xi}$$

or,

$$w(0) = \frac{\lambda}{\alpha} e^{-\gamma T} (\gamma + \xi) \quad (A3)$$

A trial of the form we have assumed is such that if the weight is the value given by Equation A3 at its commencement, then the weight will return (asymptotically) to this value after the trial. The weight can change during the trial, however. When viewed at the trial level, Equation A3 gives the asymptotic associative strength of the CS. It depends on the CS strength α , the UCS strength λ ,

the ISI length T , and the characteristics γ and ξ of the traces. For more general types of trials, the asymptotic associative strength will also depend on the durations and shapes of the CS and UCS.

APPENDIX B

A FORMAL DESCRIPTION OF THE MODEL SIMULATED IN SECTION 4

In the following, \mathbb{R} denotes the real numbers, \mathbb{R}^+ denotes the non-negative reals, and $[0,1]$ denotes the closed real interval.

Components:

one adaptive element

n plastic pathways labeled 1,...,n

1 fixed input pathway labeled 0

Descriptive Variables:

Input variables:

For each i , $0 \leq i \leq n$, $x_i(t) \in \mathbb{R}$ denotes input level on input pathway i .

Output variables:

$y(t) \in [0,1]$ denotes the output level of the adaptive element.

State variables:

$\bar{y}(t) \in [0,1]$ is called the element's expectation,

or expected output level.

For each i , $0 \leq i \leq n$, $w_i(t) \in R$ denotes the transmission efficacy or connection weight of input pathway i .

For each i , $1 \leq i \leq n$, $\bar{x}_i(t) \in R^+$ denotes the eligibility of input pathway i .

Interaction equations:

$$1) w_i(t + 1) = w_i(t) + c[y(t) - \bar{y}(t)]\bar{x}_i(t)$$

$$2) \bar{x}_i(t + 1) = \alpha\bar{x}_i(t) + x_i(t)$$

$$3) y(t) = \sum_{i=0}^n w_i(t)x_i(t) \quad (\text{bounded to remain in } [0,1])$$

$$4) \bar{y}(t + 1) = \beta\bar{y}(t) + (1 - \beta)y(t)$$

$$5) w_0(t) = w_0.$$

Parameters:

In all simulation experiments, $n = 4$, $\beta = 0$. The other parameters change from experiment to experiment (see below).

In all simulation experiments, rectangular pulse CS_is and UCSSs were represented as amplitude 1 rectangular pulses in x_i and x_0 respectively. A low level of normally distributed pseudo-randomly generated

noise (mean = .005, standard deviation = .03) was then added into the rectangular pulses. Pulse lengths varied from experiment to experiment (see below). The intertrial interval was usually 50 time steps, except where otherwise noted.

Simulation experiment particulars, by figure number in which results appeared:

Figure 4.12:

$c = .2$; $\alpha = .9$; $w = .6$
CS duration was 3 time steps; UCS duration was 30
time steps.

Figure 4.14:

$c = .5$; $\alpha = .6$; $w = .6$
 CS_2 duration was 10 time steps in trials 21-35.

Figure 4.16:

$c = .1$; $\alpha = .6$; $w = .6$ alternating with .4

Figure 4.17:

$c = .2$; $\alpha = .6$; $w = .6$; $w(0) = .6$

Figure 4.18:

$c = .1$; $\alpha = .6$; $w = .6$; $w(0) = .6$

APPENDIX C
ADAPTATION OF LEARNING RATE PARAMETERS

C.1 Preface

The work presented in this appendix is directed toward developing an algorithm for adjusting the learning rate parameter c of each synapse individually. Consider a single synapse in one of the learning elements, such as a Widrow-Hoff element or the "classical conditioning" element discussed in Section 4. This synapse is trying to use the information in its presynaptic signal to contribute to the prediction of subsequent input. One problem is that all the other synapses will also be trying to do this. If each changed itself independently of the others so that its contribution would make up the difference or error in prediction, then the next time the situation occurred, there would probably be a huge overshoot as the hundreds of active synapses each provided enough to correct the original error. In this sort of situation each synapse must proceed cautiously, changing its weight but little to prevent

overshoot, yet not so little as to make learning unnecessarily slow (undershoot).

A second and similar problem is that the signal may only provide information in a statistical sense; i.e., its presence may indicate that the input will probably be higher (or lower), but not that it definitely will be. In this case the synapse must average out the cases in which the synapse is right and wrong to arrive at a compromise measure combining both the size of the change in input predicted and the probability with which it is predicted. Again, this averaging means a slowing in the learning rate for the synapse, which must be counterbalanced against the need for speedy learning (which requires a high learning constant). How then is this learning constant to be set?

The above discussion suggests the general form of the answer: Each synapse can determine from its local measure of success in prediction - its overshoots or undershoots - whether its learning rate is too large or too small. Thus, each synapse should set its learning rate parameter as the adaptation proceeds, according to some iterative algorithm. The work presented in this appendix is the beginning of the search for, and formalization of, that algorithm.

It should be clear from the discussion of the problem

facing the individual synapse that it is basically a tracking task. The synapse is trying to track the actual input with its prediction of that input by changing its prediction proportionally to the difference between predicted and actual input in those cases in which the synapse is involved, i.e., in those cases in which the synapse is presynaptically active. In the terminology of servo-mechanism tracking, that constant of proportionality, the learning rate constant, is known as the gain. Thus, this appendix considers the problem of setting the gain of a simple tracking servo-mechanism. It is felt that the results are highly relevant to the learning rate parameter setting problem for synapses, but the work has not yet progressed to the point where it can be directly translated into this form. Further work is necessary both on the abstracted tracking problem and on mapping the results back into a learning rate parameter adaptation algorithm for a neuron-like adaptive element.

The rest of this appendix was originally a self contained paper entitled "A Method for the Automatic Selection of Gain for Discrete-Time Algorithms."

C.2 Introduction

Consider a one-dimensional, discrete-time tracking problem and its solution by a simple servomechanism (see Figure C.1). The pursuing function $y(t)$ and the target function $Y(t)$ are related according to the classic servomechanism equation:

$$y(t+1) = y(t) + G [Y(t) - y(t)] , \quad (C.1)$$

where G is called the gain. In general, the target function $Y(t)$ and the gain G will determine the quality of performance. If $Y(t+1)$ is determined from $Y(t)$ by the addition of a random variable chosen according to a symmetric probability distribution with an expected value of zero, then the optimal gain will be $G=1.0$, since then $y(t)$ will equal $Y(t-1)$, the best guess for $Y(t)$. If the target function Y has inertia, the optimal gain will lie between 1.0 and 2.0, and if $Y(t)$ is a noise corrupted version of an inertialess function $z(t)$, then the optimal gain will lie between 0 and 1.0. In this context the problem considered in this appendix is the automatic selection of a gain parameter through experience with attempts to track a target function Y .

An adaptive tracking system should have both the

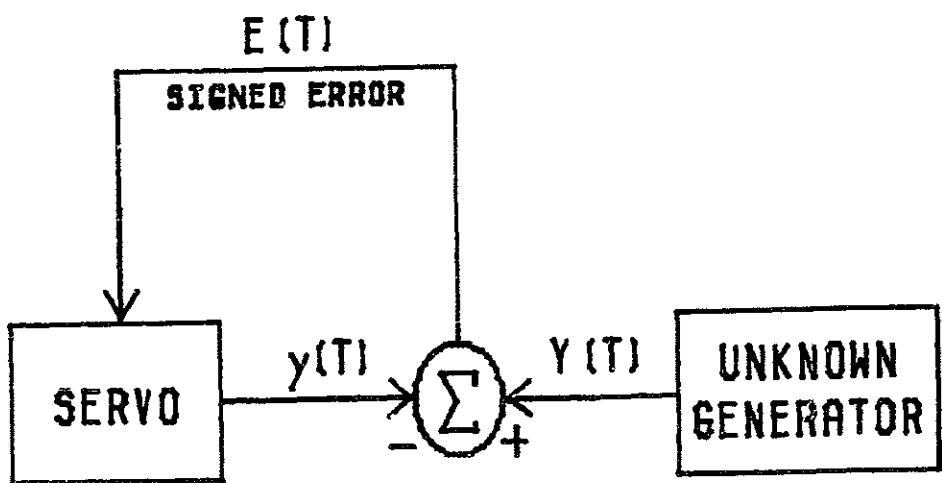


FIGURE C.1. A block diagram of a simple tracking servomechanism. $y(t)$ is the pursuing function, $Y(t)$ the target function, and $E(t)$ the signed error.

property of refinement, meaning the ability to carefully zero in on the target function by averaging out noise, and the property of responsiveness, meaning the ability to stop converging and follow the target closely if it begins to move rapidly. To have both of these properties in a tracking servomechanism requires a method of adaptively modifying the gain. Previous work on this problem apparently has not found a satisfactory solution (e.g., Eisenstein, 1972).

C.3 The Gradient Descent Approach

To optimize some parameter or vector $C(t)$ according to some evaluation function $J(t)$ to be minimized, a straightforward approach is that of gradient descent with fixed increment:

$$C(t+1) = C(t) - \alpha \nabla J(t)$$

where α is the fixed positive increment size. Ideally, one can analytically compute an expression for the gradient to get the desired algorithm. For example, this technique can be used to derive the servo equation (Equation C.1). Here

the parameter to be optimized is $y(t)$, the evaluation function $J(t)$ to be minimized is $[Y(t)-y(t)]^2$, and the positive increment is $G/2$:

$$\begin{aligned} y(t+1) &= y(t) + G/2 \nabla J(t) \\ &= y(t) + G/2 \frac{d [Y(t)-y(t)]}{d y(t)} \\ &= y(t) + G [Y(t)-y(t)] \end{aligned}$$

Yielding the servo-mechanism equation (Equation C.1).

Now let us apply the same methodology to derive an algorithm for optimizing the gain term G which we now vary as a function of time:

$$y(t+1) = y(t) + G(t+1) [Y(t)-y(t)]$$

$$\begin{aligned} G(t+1) &= G(t) - a \nabla J(t) \\ &= G(t) - a \frac{d}{dG(t)} [Y(t)-y(t)]^2 \\ &= G(t) - a \frac{d}{dG(t)} \{ Y(t) - y(t-1) + G(t)[Y(t-1)-y(t-1)] \}^2 \\ &= G(t) + 2a [Y(t)-y(t)] [Y(t-1)-y(t-1)] \\ &= G(t) + b E(t) E(t-1) \end{aligned} \tag{C.2}$$

for $b = 2a$ and $E(t) = Y(t) - y(t)$.

The intuition behind the workings of this algorithm is fairly straightforward: If the gain is too large, there will be a tendency for the pursuing function $y(t)$ to overshoot the target, which causes oscillation in the error, and thus via this algorithm will cause a decrease in the gain. If the gain is too small, on the other hand, then the pursuer will tend to undershoot, and successive errors will usually be of the same sign, and this algorithm will cause the gain to decrease. Previous approaches to this problem and its relatives have been based only on the signs of the successive errors, completely ignoring the sizes of the errors (Kesten, 1958; Sardis, 1970; Perel'man, 1967). That the algorithm presented here utilizes more of the information available in the successive errors suggests that it may be an improvement over these earlier methods.

C.4 Analysis of a Special Case

For the purposes of analysis, we now consider a special case of the general problem. Assume $Y(t)$ is a noise corrupted version of a random variable $z(t)$, and that $z(t)$

is varying as in a "random walk":

$$Y(t) = z(t) + B(t) \quad (C.3)$$

$$z(t+1) = z(t) + A(t), \quad (C.4)$$

for movement and noise random variables $A(t)$ and $B(t)$. Let us assume that the random variables $A(t)$ and $B(t)$ are chosen according to normal probability distributions with zero means and variances s_A and s_B respectively. (Since the movement of z is an inertialess random walk, for this special case the optimal gain will never be greater than 1.0.) For this case, we can prove that algorithm (C.2) converges to the gain that minimizes the expected mean square error $\text{EXP}\{[Y(t)-y(t)]^2\}$. The proof has two main steps: 1) find an expression for the optimal gain in terms of s_A and s_B , and 2) proves that Equation C.2 converges to that optimal gain. To find an expression for the optimal gain, first we find an expression for the expected asymptotic mean square error (MSE) in terms of s_A , s_B , and the gain G .

Let $e(t) = z(t) - y(t)$

Then note that the total error can be written

$$E(t) = e(t) + b(t). \quad (C.5)$$

Now we solve for asymptotic $e(t)$:

$$\begin{aligned} e(t+1) &= z(t+1) - y(t+1) \\ &= z(t) + A(t) - y(t) - G E(t) \\ &= e(t) + A(t) - G [e(t) + B(t)] \\ &= (1-G)e(t) + A(t) - G B(t) \end{aligned}$$

or

$$e(t) = (1-G) e(0) + \sum_{n=0}^{t-1} (1-G)^n [A(t-n) - G B(t-n)]$$

Let $e(\infty)$ denote the limit of this expression as t goes to infinity. Since $e(\infty)$ is a sum of independent identically normally distributed random variables, it will also be normally distributed, will have mean zero, and will have variance the sum of the variances of the summands:

$$s^2_{e(\infty)} = \lim_{t \rightarrow \infty} \sum_{n=0}^t s^2_{\{(1-G)^n [A(t-n) - G B(t-n)]\}}$$

$$= \lim_{t \rightarrow \infty} \sum_{n=0}^t [(1-G)^2]^n [s^2_A + G^2 s^2_B]$$

where s_x^2 denotes the variance of the random variable X .

This geometric series is convergent for $0 \leq G \leq 2.0$:

$$s^2_{e(\infty)} = \frac{s^2_A + G^2 s^2_B}{1 - (1-G)^2}$$

By (C.5), and since $e(\omega)$ is normally distributed with mean zero, $E(\omega)$ is also normally distributed with mean zero and of variance

$$\frac{s^2}{E(\omega)} = \frac{sA^2 + G^2 sB^2}{1 - (1-G)} + sB^2 \quad (C.6)$$

Which is just the desired equation for the mean square error in terms of sA , sB , and G . The value of G which minimizes this MSE can be found by the straightforward but tedious process of differentiating Equation C.6 with respect to G and setting it to zero. After simplification and solving a quadratic, a single positive root is found:

$$G_{\text{opt}} = \frac{-sA^2 + \sqrt{sA^4 + 4 sA^2 sB^2}}{2 sB^2} \quad (C.7)$$

For the second part of the proof we must show that Equation C.2 converges to the optimal gain (C.7). From (C.2) and (C.6):

$$G(t+1) = G(t) + b E(t) E(t-1) \quad (C.8)$$

We will assume that if the constant b is chosen properly, $G(t)$ will (nearly) converge to the fixedpoint of (C.8), and only prove that that fixedpoint is (C.7). (Note: to really complete the convergence proof it is necessary to let the

increment b become an decreasing sequence and prove a contraction property on the expected change in $G(t)$ as t goes to infinity.) At the fixedpoint G of (C.8)

$$\begin{aligned}
 0 &= \text{EXP}\{ E(t+1) E(t) \} \\
 &= \text{EXP}\{ [e(t+1) + B(t+1)] E(t) \} \\
 &= \text{EXP}\{ [(1-G)E(t) - B(t) + A(t) + B(t+1)] E(t) \} \\
 &= \text{EXP}\{ (1-G)E(t)^2 - E(t)B(t) + E(t)A(t) + E(t)B(t+1) \} \\
 &= (1-G)\text{EXP}\{E(t)^2\} - \text{EXP}\{[e(t)+B(t)] B(t)\} \\
 &= (1-G)\text{EXP}\{E(t)^2\} - \text{EXP}\{B(t)^2\} \\
 \\
 &= (1-G) \frac{s^2}{E(\infty)} - sB^2
 \end{aligned}$$

Substituting in with (C.8), and simplifying yields

$$0 = sB^2 G^2 + sA^2 G - sA^2,$$

whose only positive root is the same as (C.5), the expression for the optimal gain.

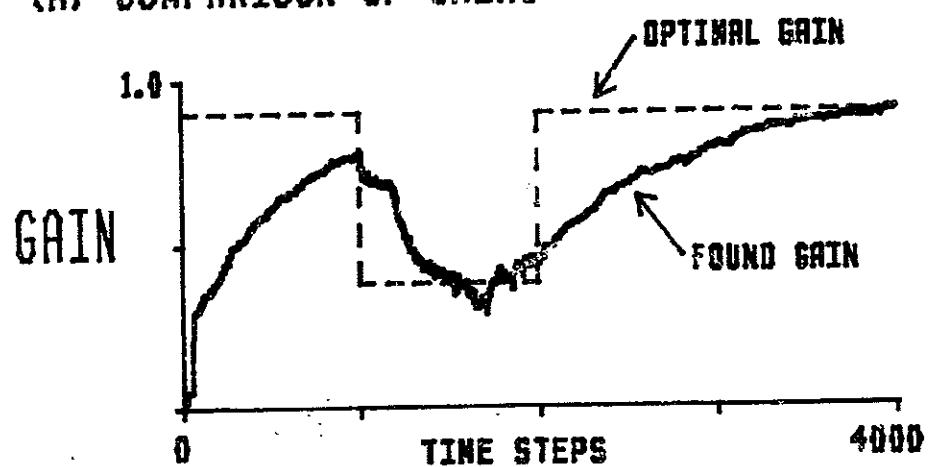
C.5 Computer Simulation

The algorithm (Equations C.1, C.2, C.3, C.4) was programmed on a digital computer, with the distributions of the random variables approximated by pseudo-random number generating programs. Figure C.2 reports the results of an experiment in which the observation noise standard deviation s_B was step changed from $s_B=0.3$ to $s_B=2.0$ and back again. Figure C.2a shows the optimal gain compared to the actual gain, both versus time. This figure demonstrates that the gain adaptation algorithm can both increase and decrease the gain, whichever is appropriate. Figure C.2b shows the analytic asymptotic error for the actual and optimal gains plotted versus time. This figure illustrates that the algorithm can keep the error of the tracking system at very nearly the optimal theoretical limit despite occasional or slow changes in the unknown system and thus in the optimal gain.

C.6 Further Levels of Adaptation

One nice aspect of the algorithm presented here is that only one parameter, the gain increment parameter b , need be

(A) COMPARISON OF GRINS



(B) COMPARISON OF ASYMPTOTIC ERRORS

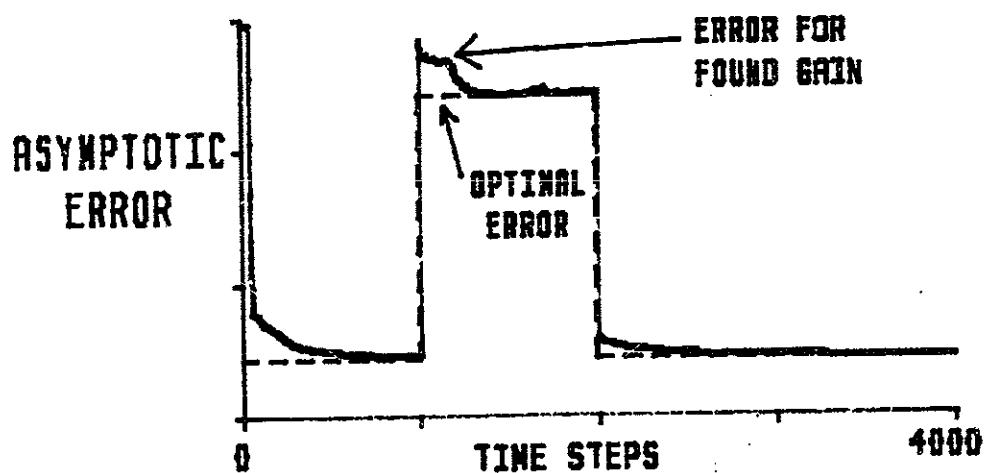


FIGURE C.2

FIGURE C.2. Computer simulation of the single level adaptive gain selection algorithm. In this experiment the observation noise standard deviation parameter was step changed from $sB=0.3$ to $sB=2.0$ and then back again while the random movement standard deviation parameter remained constant at 1.0. These changes were made at the 1000th and 2000th time steps respectively.

FIGURE C.2a compares the analytic optimal gain (dashed line) with that found by the single-level gain adaptation algorithm (solid line). Note that the algorithm can both increase and decrease the gain.

FIGURE C.2b compares the analytic asymptotic error (MSE) levels under the optimal (dashed line) and actual gains (solid line). The changes in actual gain keep the error nearly at the theoretical minimum despite the changes in the observation noise. The gain change rate parameter was $b=0.001$ in this experiment.

chosen arbitrarily by the designer or user of the technique to fit the characteristics of the particular application. This is in sharp contrast to the methods of Perel'man (1967) and Kesten (1958), whose performance is dependent on a series of possible gain parameters that need to be specified by the user. In the algorithm here, even the dependence on the b parameter can be reduced, i.e., can be made automatically adaptive to the environment, by extending the scheme to additional levels of adaptation. Applying the same methodology we used twice above, we let b become a function of time and change it in proportion to the gradient of the evaluation function $J(t)$ with respect to $b(t)$:

$$b(t+1) = b(t) - \alpha \frac{\nabla J(t)}{b(t)}$$

Solving this analytically results in an algorithm for the optimal rate of change of gain parameter. This algorithm will in turn have a rate parameter, and an optimizing algorithm can be derived for that. The result is an arbitrarily deep hierarchy of rate of change or gain algorithms. A pattern in these algorithms quickly becomes apparent. We change notation slightly at this point to allow a statement of the multiple-level adaptive gain selection scheme which makes this pattern more apparent. For a gain selection algorithm with n levels of adaptation:

$$y(t+1) = y(t) + \sum_1^n G_i(t) E_i(t)$$

$$\text{where } E_i(t) = Y(t) - y(t)$$

$$G_i(t+1) = G_i(t) + G_{i+1}(t) E_i(t) \quad i=1, \dots, n-1$$

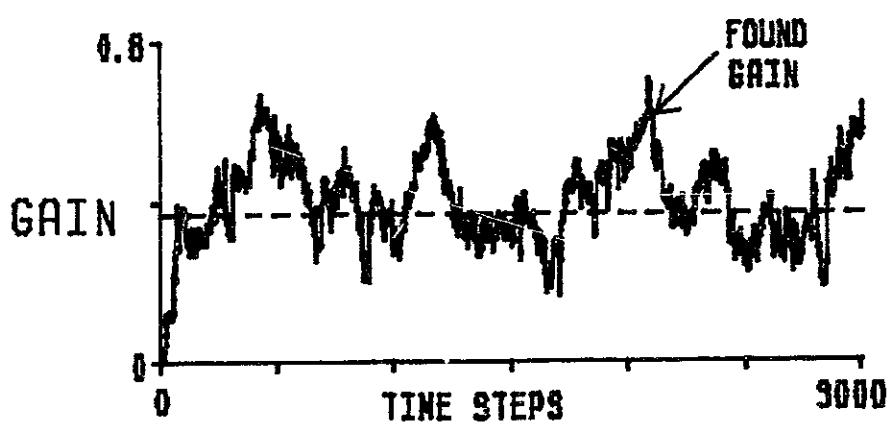
$$\text{where } E_i(t) = E_{i-1}(t) E_{i-1}(t-1) \quad i=1, \dots, n-1$$

and $G_n(t+1) = G_n$, a small positive constant for the last

level of adaptation.

This multiple-level algorithm was also programmed on a digital computer for the special case of normally distributed movement and noise random variables. An experiment was run comparing the previous two level system (the first level of adaptation was just the simple servo itself) to a three level system for a case in which the optimal gain remained constant. We see from Figure C.3a that while the two level system found the optimal gain very quickly, there was no tendency for the gain to converge to that value. The three level system, on the other hand, was able to detect that the optimal gain itself was not changing, and reduced the rate at which it changed the gain, resulting in the convergence of the gain to its optimal value (Figure C.3b). However, simulation results also revealed that the multiple-level algorithm can become

(A) SINGLE LEVEL ALGORITHM



(B) TWO LEVEL ALGORITHM

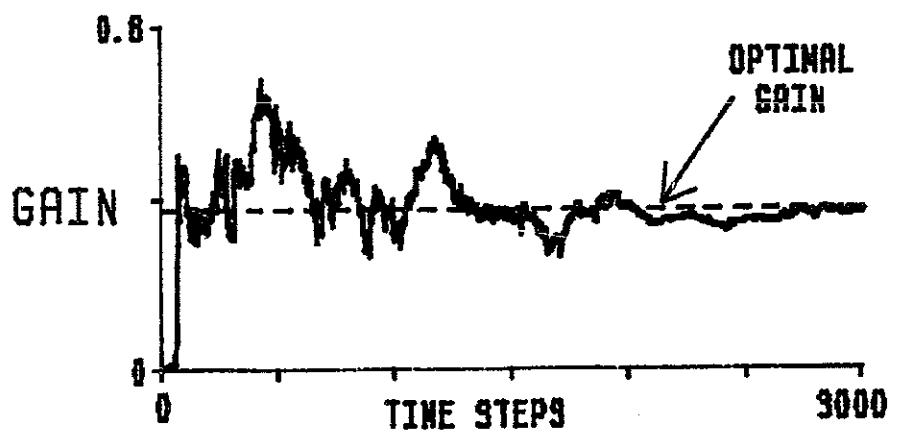


FIGURE C.3

FIGURE C.3. Comparison of a single level gain adaptation algorithm with a two level algorithm for the case of a constant optimal gain. While the single level system (Figure C.3a) finds the optimal gain (dashed line), there is no tendency for the gain to converge to it. The two level gain adaptation algorithm (Figure C.3b), on the other hand, can adjust the rate with which it varies its gain, and does converge to the optimal gain. In this experiment $sA=1.0$, $sB=3.0$, and the rate or gain constants for the last levels of adaptation were $\text{gain}(2)=0.001$ for Figure C.3a and $\text{gain}(C.3)=5.0e-8$ for Figure C.3b.

unstable for some unknown systems and for some settings of the rate or gain constants for the last level of adaptation. This is probably due to the fact that the gradient descent analysis technique is due to a linear approximation of the gradient of the evaluation function $J(t)$. If the increment is small, this approximation is a good one, but if the increment is large, it can be a very poor approximation to the actual gradient. In the multiple-level algorithm this increment is under adaptive control, and thus there is no guarantee that the increment will remain sufficiently small, and instability can result. Further work is needed to solve this problem with the otherwise promising multiple-level algorithm.

C.7 Conclusions

The multiple-level gain selection algorithm presented here seems to be applicable to any case of discrete-time adaptation involving a signed error and an associated gain or rate parameter. This algorithm is able to both increase and decrease gain in response to changes in the target function's behavior, utilizes all the information in the error signal, and is extremely simple. Comparisons are difficult to make between dissimilar algorithms, but the above

properties suggest that this algorithm may be a significant improvement over other gain or rate parameter selection algorithms in the literature.

Finally, a multiple-level version of this algorithm was presented. Its particular advantages will be most important in systems which must handle with high performance a wide range of uncertain environments. Although the approach seems promising, further work is necessary on the multiple-level algorithm.

APPENDIX D

DETAILS OF THE SIMULATION EXPERIMENTS OF SECTION 7

D.1 Computation of movement

At each time step the simulated adaptive network provides an instantaneous action vector $\{A[\text{right}](t), A[\text{left}](t)\}$. The computation of this vector is detailed in Appendix E. A record $SA[a](t)$ is kept of the extent to which each action a has been instantaneously selected recently:

$$\begin{aligned} SA[\text{right}](t) &= \alpha * SA[\text{right}](t-1) + (1-\alpha) * A[\text{right}](t) \\ SA[\text{left}](t) &= \alpha * SA[\text{left}](t-1) + (1-\alpha) * A[\text{left}](t) \end{aligned}$$

Movement is determined by which of these traces is largest:

$$\text{Motion}(t) = \beta * \{SA[\text{right}](t) - SA[\text{left}](t)\},$$

where positive motion means motion to the right, and negative motion means motion to the left. In all of the simulation experiments the constants α and β were set at 0.8 and 50.0 respectively. If the motion computed above causes the subject to run into a barrier, the actual motion is halted at the point of contact. In addition, barrier collision neutralizes the inertial tendency to continue motion in that direction. Specifically, the inertial traces $SA[\text{right}]$ and $SA[\text{left}]$ are set to their average upon collision with a barrier. The inertia was also neutralized by setting both of these traces to zero each time a subject was "picked up" and moved as part of an experiment.

DETAILS OF THE SIMULATION EXPERIMENTS OF SECTION 7 PAGE D-2

D.2 Experiment I

Experiment 1 used 200 subjects, each run individually through the following three phases.

D.2.1 Exploration Phase

Each subject was released between the lower large colored regions (point A in Figure 7.4). If the center of its body passed into a colored region, the corresponding sensory input line was set to a value of approximately 0.5. All motion was computed as described above. After 1000 time steps the association phase began.

D.2.2 Association Phase

Each subject was moved to the enclosed red region D of Figure 7.4. The red input line was activated in the same way it was activated during the exploration phase when the subject was within the lower red region. After two time steps the reward input line was also set to 0.5. After one time step of this stimulation pattern, each subject was transferred to the enclosed green box marked B in Figure 7.4. The input pattern there was $I[\text{red}]=0.0$, $I[\text{green}]=0.5$, and $I[\text{reward}]=0.5$. After two time steps of this, the reward input line was set to zero again for one time step, and then the testing phase began. The following chart summarizes the stimulation regime during the association phase.

absolute time	duration	$I[\text{red}]$	$I[\text{green}]$	$I[\text{reward}]$
1000-1001	2	0.5	0.0	0.0
1002	1	0.5	0.0	0.5
1003-1004	2	0.0	0.5	0.5
1005	1	0.0	0.5	0.0

DETAILS OF THE SIMULATION EXPERIMENTS OF SECTION 7 PAGE D-3

D.2.3 The Testing Phase

In the testing phase of the primary experiment each subject was returned to location A of Figure 7.4 and released, just as in the exploration phase. The testing phase ended when either of the two colored regions was entered. Of the 200 subjects, 141 entered the red region first and 59 entered the green region first. This result is statistically significant to at least the $P=.005$ level.

D.3 Experiment II

The second experiment was identical to the first during the exploration and association phases. Its testing phase differed in that the lower red and green regions and the barriers inside them were removed. After 300 time steps the testing phase ended and the position of the subject was recorded. All of the 100 subjects had moved very far to the right after the 300 time steps, the nearest being about twice as far off the page as the distance from A to the right edge of the page in Figure 7.4.

D.4 Experiment III

The third experiment was identical to the first experiment except that the bias weights WAC[right] and WAC[left] (also called B[right] and B[left] in the text) were set to zero at the beginning of the testing phase. This ensured the subjects had no initial tendency to move either right or left at the beginning of the testing phase. Of the 100 subjects, 71 entered the red area first, a result statistically significant at least the $P=.005$ level.

APPENDIX E

DETAILS OF THE ADAPTIVE NETWORK SIMULATED IN SECTION 7

Notation: \mathbb{R} is the Reals, \mathbb{R}_+ the positive Reals
 \in means "element of"
STIMULI is the set {RED, GREEN, REWARD}
ACTIONS is the set {RIGHT, LEFT}

This is a discrete time model, i.e. $t=0, 1, 2, \dots$

E.1 Components

A PREDICTOR-MODULE, consisting of 3 PREDICTOR-ELEMENTS (corresponding to the 3 stimuli), a 3×3 matrix of PREDICTOR-TO-PREDICTOR-CONNECTIONS, and a 3×2 matrix of ACTOR-TO-PREDICTOR-CONNECTIONS.

An ACTION-SELECTING-MODULE, consisting of 2 ACTOR-ELEMENTS and a 2 element vector of CONSTANT-TO-ACTOR-CONNECTIONS.

A vector of 3 INPUT-LINES, corresponding to the three STIMULI.

A vector of two OUTPUT-LINES, corresponding to the 2 ACTIONS.

E.2 Descriptive Variables

E.2.1 Input Variables

$I[s](t) \in [0,1]$, for all $s \in \text{STIMULI}$, is the input to the network at time t . These indicate the color of the region the subject is in (if any) and the presence or absence of reward.

E.2.2 Output Variables

$A[a](t) \in [0,1]$, for all $a \in \text{ACTIONS}$, is the activity level at time t of the ACTOR-ELEMENT for action a , indicating the instantaneous selection of movement to the right or left.

E.2.3 State Variables

$P[s](t) \in [0,1]$, for all $s \in \text{STIMULI}$, is the activity level at time t of the PREDICTOR-ELEMENT for stimulus s . This indicates a combination of prediction of stimulation and actual stimulation.

$WPP[s_1,s_2](t) \in R$, for all $s_1, s_2 \in \text{STIMULI}$, is the efficacy of the PREDICTOR-TO-PREDICTOR-CONNECTION to the PREDICTOR-ELEMENT for stimulus s_1 from the PREDICTOR-ELEMENT for stimulus s_2 .

$WPA[s,a](t) \in R$, for all $s \in \text{STIMULI}$, $a \in \text{ACTIONS}$, is the efficacy at time t of the ACTOR-TO-PREDICTOR-CONNECTION to the PREDICTOR-ELEMENT for stimulus s from the ACTOR-ELEMENT for action a .

$WAC[a](t) \in R$, for all $a \in \text{ACTIONS}$, is the efficacy at time t of the CONSTANT-TO-ACTOR-CONNECTION to the ACTOR-ELEMENT

for action a. These weights were called the bias weights and denoted $B[a]$ rather than $WAC[a]$ in the text.

$TA[a](t) \in [0,1]$, for all $a \in$ ACTIONS, is the trace at time t of $A[a](t)$, the activity of the ACTOR-ELEMENT for action a.

$TA2[a](t) \in [0,1]$, for all $a \in$ ACTIONS, is another trace at time t of $A[a](t)$.

$TP[s](t) \in [0,1]$, for all $s \in$ STIMULI, is the trace at time t of $P[s](t)$, the activity of the PREDICTOR-ELEMENT for stimulus s.

E.2.4 Parameters:

$CPP[s_1, s_2] \in R_+$, for all $s_1, s_2 \in$ STIMULI, is the learning rate parameter for the PREDICTOR-TO-PREDICTOR-CONNECTION from the PREDICTOR-ELEMENT for stimulus s_2 to the PREDICTOR-ELEMENT for stimulus s_1 .

$CPA[s, a](t) \in R_+$, for all $s \in$ STIMULI, $a \in$ ACTIONS, is the learning rate parameter for the ACTOR-TO-PREDICTOR-CONNECTION from the ACTOR-ELEMENT for action a to the PREDICTOR-ELEMENT for stimulus s.

$CAC[a] \in R_+$, for all $a \in$ ACTIONS, is the learning rate parameter for the CONSTANT-TO-ACTOR-CONNECTION to the ACTOR-ELEMENT for action a.

Mean \bar{R} , Stdev R_+ are the mean and standard deviation parameters for the normally distributed noise component of the activity of the ACTOR-ELEMENTS.

$Ap \in R$ is the trace decay parameter for the trace of activity in the PREDICTOR-ELEMENTS.

$Aa \in R$ is the trace decay parameter for the trace of activity in the ACTOR-ELEMENTS that is used to changes the ACTOR-TO-PREDICTOR-CONNECTION efficacies.

$Aa2 \in R$ is the trace decay parameter for the trace

of activity in the ACTOR-ELEMENTS that is used to changes the CONSTANT-TO-ACTOR-CONNECTION efficacies.

E.3 Equations of Interaction

E.3.1 Equations of primary network operation:

$$\begin{aligned} A[\text{right}](t) &= f\{ A'[\text{right}](t) - A'[\text{left}](t) \} \\ A[\text{left}](t) &= f\{ A'[\text{left}](t) - A'[\text{right}](t) \} \end{aligned}$$

where $A'[a](t) = \text{MAX}\{0, WAC[a](t) + \text{NOISE}[\text{mean}, \text{stdev}]\}$
for all a ACTIONS and $f[x] = \text{MAX}\{0, \text{MIN}\{x, 1.0\}\}$,
and $\text{NOISE}[\text{mean}, \text{stdev}]$ is a normally distributed random variable.

$$P(t) = f\{ I(t) + WPA(t) A(t) + WPP(t) P(t-1) \}$$

(using vector and matrix notation)

E.3.2 Equations for change of connection efficacies:

$$WPP[s_1, s_2](t+1) = WPP[s_1, s_2](t) + CPP[s_1, s_2] * \{ P[s_1](t) - TP[s_1](t) \} * TP[s_2](t-1)$$

$$WPA[s, a](t+1) = WPA[s, a](t) + CPA[s, a] * \{ P[s](t) - TP[s](t) \} * TA[a](t)$$

$$WAC[a](t+1) = WAC[a](t) + CAC[a] * \{ P[\text{reward}](t) - TP[\text{reward}](t) \} * TA2[a](t)$$

where:

$$\begin{aligned} TP[s](t+1) &= Ap * TP[s](t) + (1.0 - Ap) * P[s](t) \\ TA[a](t+1) &= Aa * TA[a](t) + (1.0 - Aa) * A[a](t) \\ TA2[a](t+1) &= Aa2 * TA2[a](t) + (1.0 - Aa2) * A[a](t) \end{aligned}$$

For all $a \in$ ACTIONS and $s, s_1, s_2 \in$ STIMULI.

E.4 Parameter Settings:

CPP[s ₁ , s ₂]	\	s ₁			
	s ₂	\	RED	GREEN	REWARD
	REWARD		0.5	0.5	0.0
	GREEN		0.5	0.0	1.5
	RED		0.0	0.5	1.5

CPA[s, a]:	\	s			
	a	\	RED	GREEN	REWARD
	RIGHT		0.2	0.2	0.0
	LEFT		0.2	0.2	0.0

$$CAC[right] = 0.5 \quad Ap = 0.0$$

$$CAC[left] = 0.5 \quad Aa = 0.8$$

$$Aa2 = 0.0$$

$$Mean = 0.2$$

$$Stdev = 0.4$$

APPENDIX F

DESIGNNET: NETWORK SIMULATION DISPLAY PACKAGE

F.1 Description

DESIGNNET is a collection of routines for interactively creating a network display on the Grinnell and then using the display to show the values of variables associated with the network. The network resembles a neural structure consisting of sets of neurons with input and output fibers and weighted connections (synapses). At most three variables can be displayed at each cell and at each synapse. Their values are shown as color intensity, the radius of a circle, or the width of a square. The values of the variables are assigned by the user's program; the values are not changed by DESIGNNET.

F.2 User Instructions

The user instructions are presented in two sections: the creation of a network display, and the subsequent use of the display.

F.2.1 Creating a Network Display

A network display can be created or modified by running the DESIGNNET program as follows:

```
RUN DR1:[ANWCA.DNET]DESIGNNET
```

You will then be asked to type a name for the file in which you will save your network, or for a file of an existing network that you wish to modify.

FILENAME FOR SAVING AND RETRIEVING?
example.net

From this point on you will be interacting with the program through the Grinnell. Various options will appear in a list on the right side of the screen. Any one of these options can be selected by moving cursor one into the box surrounding it and pressing the enter button. Some of the options will present a new list of options to select from. The previous option list can be returned to by pressing home, then enter (cursor one must be on).

Initial Options and Descriptions

REFRESH

Redraws the entire network display.

EDIT

Allows the creation and modification of a network display. This option requires the selection of further options that are described below.

SAVE NET

Saves the current network display in the file specified by the file name entered when DESIGNNET was started, or the file name entered in response to the option SET FILE.

RETRIEVE NET

Replaces the current network display with the network that was previously saved in the file specified by the file name entered when DESIGNNET was started, or the file name entered in response to the option SET FILE. This option will erase the current network so a safety feature requires this option to be selected a second time to continue.

SET FILE

Allows you to enter the name of a file to be referenced when saving and retrieving networks with the SAVE NET and RETRIEVE NET options.

UPDATE NET

This option will operate only if an UPDATE_NET_DISP and a BRAIN_RETRIEVE routine have been compiled and linked to DESIGNNET. BRAIN_RETRIEVE must read the data stored in the BRAIN data file. The I/O unit number for the BRAIN data file is passed to BRAIN_RETRIEVE. UPDATE_NET_DISP must call UPDATE_MOD_DISP or UPDATE_SHORTMOD to display the variables retrieved by BRAIN_RETRIEVE.

UPDATE_NET_DISP has no arguments. The documentation for the EXPER program explains the use of BRAIN_RETRIEVE and the BRAIN data file.

MOD NUMBERS

This option displays a number on each module of the network, representing their respective module numbers. These numbers must be known when you call the various entry points for updating the network display. (see section 2.2)

EDIT Option List and Descriptions

ADD SHAPE

This will add a new shape to the display. Shapes can be used to enhance the display by emphasizing certain areas, highlighting lettering, etc. (The maximum number of shapes is 20.)

First ADD SHAPE Options:

RECTANGLE

The added shape will be a rectangle.

SYMBOL

Not functional at this time.

Second ADD SHAPE Options:

POSITION

Use one or two cursors (depending on the shape) to position the shape and to specify its size.

COLOR

Allows you to select the shape's color by adjusting the intensities of red, green, and blue with cursor one.

ADD STRING

This will add a new string of characters to the display. The characters must be typed on the terminal when requested to do so. Strings of characters can be used to label parts of a network or as titles. (The maximum number of strings is 20, and the maximum number of characters per string is 70.)

ADD STRING Options:

POSITION

Use cursor one to position the character string.
Cursor one is at the lower left corner of the
character string.

HEIGHT

Changes the height of the characters from single
to double height, or vice versa. Single height
is 9 pixels, and double height is 18 pixels.

WIDTH

Changes the width of the characters from single
to double width, or vice versa. Single width is
7 pixels, and double width is 14 pixels.

ANGLE

Changes the angle of the character string in 45
degree increments. The character string is
rotated around the first character of the
string.

TEXT

Requests you to type a new string of characters
at the terminal to replace the old character
string.

COLOR

Allows you to select the color of the characters
by adjusting the intensities of red, green, and
blue with cursor one.

ADD MODULE

A module is a set of cells and fibers or just fibers.
There are three types of modules, one of which must be
selected in the next option list. (The maximum number
of modules is 15.)

First ADD MODULE Options:

STANDARD

This type of module consists of cells arranged
in a line with parallel dendrites and axons.
Input fibers, which are optional, are parallel
to each other, but perpendicular to the cell's
dendrites. The intersection of the input fibers
and dendrites represent synapses at which
several variables can be displayed. Recurrent
fibers, also optional, connect the cell's axon
with all dendrites in the module. These also
form intersections, representing synapses, with
the dendrites. After selecting this option two
numbers must be chosen as follows:

NUMBER OF CELLS

Choose the number of cells in this module by moving cursor one vertically. There can be from 1 to 10 cells in one module of this type. Push home and enter when you have chosen the desired number of cells.

NUMBER OF INPUTS

Choose the number of input fibers in this module by moving cursor one vertically. There can be from 0 to 10 input fibers in one module of this type. Push home and enter when you have chosen the desired number of input fibers.

PARALLEL

This type of module consists only of straight, parallel fibers. This module can be used to connect two other modules. After selecting this option one number must be chosen as follows:

NUMBER OF CELLS

In this case, the number of cells is actually the number of fibers. Choose the number of fibers by moving cursor one vertically. There can be from 1 to 10 fibers in one module of this type. Push home and enter when you have chosen the desired number of fibers.

CORNER

This type of module consists of parallel fibers, each with one 90 degree bend. This module can be used to connect two other modules. After selecting this option one number must be chosen as follows:

NUMBER OF CELLS

In this case, the number of cells is actually the number of fibers. Choose the number of fibers by moving cursor one vertically. There can be from 1 to 10 fibers in one module of this type. Push home and enter when you have chosen the desired number of fibers.

Second ADD MODULE Options:

POSITION

A module may be moved to any location on the screen with cursors one and two; cursor one will be one corner of the module and cursor two

DESIGNNET: NETWORK SIMULATION DISPLAY PACKAGE PAGE F-6
User Instructions

will be the opposite corner. The cursors also control the size and orientation of the module. Push enter when you have positioned both cursors.

AXON LENGTH

This allows you to extend the axons of all cells in a STANDARD module, and the fibers of the PARALLEL and the CORNER modules, by moving cursor one. Push enter when you are done.

RECURRENTY

This option affects only a STANDARD module. If the STANDARD module does not have recurrent fibers, then selecting this option will put recurrent fibers in the module, and vice versa.

FLIP

This will flip a module of STANDARD or PARALLEL type by 90 degrees, i.e., the module is reflected on the diagonal between the module's corners in which the two cursors appear during positioning. This option will not flip a CORNER module in this manner. Instead, it reverses the order of the fibers along one side of the module.

NUM VARS

This option is used to designate the number of variables to be displayed at cell bodies and at fiber intersections. These numbers are entered as follows:

NUMBER OF CELL VARIABLES

Choose the number of cell variables to be displayed with this module by selecting one of the numbers 1, 2, or 3. The number of cell variables is initially 1. Variable 1 is displayed as the intensity of the color (yellow) of the cell. Variable 2 is displayed as the radius of a white circle centered at the cell body. Variable 3 is displayed as half the width of a white square centered at the cell body.

NUMBER OF SYNAPTIC VARIABLES

Choose the number of synaptic variables to be displayed with this module by selecting one of the numbers 1, 2, or 3. The number of synaptic variables is initially 2. Variable 1 is displayed as the radius of a green or red disk, corresponding to a

positive or negative value, centered at the synapse. Variable 2 is displayed as the radius of a white circle centered at the synapse. Variable 3 is displayed as half the width of a white square centered at the synapse.

BOUNDS

This option allows you to set the minimum and maximum values of the cell and synapse variables for one module. These are used to scale the intensity or size of the variable's display. The bounds are entered as follows:

CELL VARIABLE BOUNDS

Move cursor one vertically to set the minimum and maximum values for each cell variable in turn. Push enter after each number is selected. Push home and enter when both the minimum and maximum have been chosen. The cell variable bounds are initially 0.0 to 1.0

INPUT VARIABLE

In a similar manner, set the minimum and maximum values for the one input variable. These bounds will scale the intensity of the color of the input fibers. The input variable bounds are initially 0.0 to 1.0. (The input variable bounds are needed only if this module has input fibers.)

INPUT SYNAPSE VARIABLES

In a similar manner, set the minimum and maximum values for each input synapse variable. The input synapse variable bounds are initially -0.1 to 0.1. (The input synapse variable bounds are needed only if this module has input fibers.)

RECURRENT SYNAPSE VARIABLES

In a similar manner, set the minimum and maximum values for each recurrent synapse variable. The recurrent synapse variable bounds are initially -0.1 to 0.1. (The recurrent synapse variable bounds are needed only if this module has recurrent fibers.)

MODIFY ELEMENT

This option will allow you to change any object currently in your network. You must designate which

object you want to modify by placing cursor one on it and pressing enter. Choosing a character string, however, requires cursor one to be close to the initial character of the string. If the cursor was not close to any object when enter was pressed, you must try again. After successfully choosing an object, the appropriate set of options become available. They are listed here:

Options if a SHAPE is chosen: described under main option ADD SHAPE)

POSITION
COLOR

Options if a character string is chosen: (described under main option ADD STRING)

POSITION
HEIGHT
WIDTH
ANGLE
TEXT
COLOR

Options if a module is chosen: (described under main option ADD MODULE)

POSITION
AXON LENGTH
RECURRENCY
FLIP
NUM VARS
BOUNDS

DELETE ELEMENT

This option allows you to remove an object from your network. You designate the object to be deleted by placing cursor one on it and pressing enter. A character string, however, is chosen by placing cursor one close to the initial character in the string. If an object is in a complex area of the network it might be necessary to choose it after you have moved some of the nearby elements away from the object. After deleting the object, these elements may be moved back into position. If the cursor is not close to an object when enter is pressed, you must try again.

BACK COLOR

This option allows you to change the background color. Set the intensities of red, green, and blue by moving cursor one vertically. When done with one color push enter to set the next color. When all colors are set push home and enter.

SKEL COLOR

The skeleton of a network is that part of the fibers and cells that does not change color during variable display, such as the fiber borders. This option allows the skeleton color to be changed by setting the intensities of red, green, and blue by moving cursor one vertically. When done with one color push enter to set the next color. When done with all colors push home and enter.

F.2.2 Displaying the Network from Your Program

This section describes the entry points that can be called to display a previously constructed network and dynamically update the display variables.

Your program must be linked to the library of DESIGNNET routines, the library of ANW graphics routines, and to the library of Grinnell routines, for example:

```
LINK your program,DR1:[ANWCA.DNET]DNETLIB/LIB,  
[ANWRS.GRF]GLIB/LIB,[MOVIE]GRLIB1/LIB
```

Initializing the Display

Before displaying the network your program must initialize the Grinnell. This can be done by putting the following two statements in your program before any DESIGNNET routines are called:

```
CALL GR_INITIALIZE (0, ' ')  
CALL GR_CONFIG444
```

To prepare a network for display call the subroutine

```
PREPARE_NET (filename)
```

where filename is a character string giving the name of the file in which a network has been stored. This file can be interactively generated or modified by using the DESIGNNET program described in section 2.1. PREPARE_NET does not display anything and need only be called once. To display the initial state of the network call the subroutine

```
DISP_NET_SKEL
```

which will draw all shapes, texts, and modules on the Grinnell. DISP_NET_SKEL can be called at any point to refresh the network display.

Displaying Variables

To display the current state of the network variables in one module call one of the following subroutines:

```
UPDATE_MOD DISP (module_number,
    cell_var1, cell_var2, cell_var3,
    input_var,
    rct_syn_var1, rct_syn_var2, rct_syn_var3,
    inp_syn_var1, inp_syn_var2, inp_syn_var3,
    num_cells_in_module, num_inputs_in_module)
```

or

```
UPDATE_SHORTMOD (module_number, cell_var1)
```

where the arguments are:

module_number: INTEGER variable

This is the number corresponding to the module that is to be updated.

cell_var1, cell_var2, cell_var3: REAL arrays dimensioned (num_cells_in_module)

These are the values of the 3 cell variables for each cell. Var1 is displayed as the cell's color intensity. Var2 is displayed as the radius of a circle centered at the cell body. Var3 is displayed as half the width of a square centered at the cell body.

input_var: REAL arrays dimensioned (num_inputs_in_module)

These are the values of the variable for each input fiber, displayed as the fiber's color intensity.

rct_syn_var1, rct_syn_var2, rct_syn_var3: REAL arrays dimensioned (num_cells_in_module, num_cells_in_module)

These are the values of the 3 recurrent fiber synapse variables. Var1 is displayed as the radius of a green or red disk, corresponding to a positive or negative value, centered at the synapse. Var2 is displayed as the radius of a circle centered at the synapse. Var3 is displayed as half the width of a square centered at the synapse.

inp_syn_var1, inp_syn_var2, inp_syn_var3: REAL arrays dimensioned (num_inputs_in_module, num_cells_in_module)

Displaying the Network from Your Program

These are the values of the 3 input fiber synapse variables, and are displayed in a manner similar to the rct_syn_var's.

Not all modules will require the display of all 3 variables of each type. A single variable can be used as a dummy for values that are not needed. For example, if only the first variable of each type is needed, call the subroutine with:

```
UPDATE_MOD_DISP (module_number, cell_var1, dummy, dummy,
                  input_var,
                  rct_syn_var1, dummy, dummy,
                  inp_syn_var1, dummy, dummy,
                  num_cells_in_module,
                  num_inputs_in_module)
```

If only the first cell variable is needed, then UPDATE SHORTMOD can be called. Every module of type PARALLEL and CORNER should call UPDATE_SHORTMOD.

To display the current state of the entire network UPDATE_MOD_DISP or UPDATE_SHORTMOD must be called for each module in the network.

Changing Display Parameters

There are routines available for changing the color of the network skeleton and the bounds of the variables. These routines can be called at any time.

To change the color of the network skeleton call:

```
DEF_SKEL_COLOR (red, green, blue)
```

"Skeleton" refers to the cells and fibers without the associated variables. This color is seen on the cell and fiber borders at all times and on the cells and fibers in which variable 1 is at a minimum. The arguments red, green, and blue are of type INTEGER.

The bounds of the network variables are set when the network is created, but can be changed for the duration of a program's execution. To change the bounds of the cell variables call:

```
DEF_ACT_RANGE (module_number, var_number, var_minimum,
                var_maximum)
```

To change the bounds of the input fiber variable call:

DESIGNNET: NETWORK SIMULATION DISPLAY PACKAGE PAGE F-12
Displaying the Network from Your Program

DEF_INP_RANGE (module_number, var_minimum, var_maximum)

To change the bounds of the input fiber synapse variables call:

DEF_IWGHT_RANGE (module_number, var_number, var_minimum, var_maximum)

To change the bounds of the recurrent fiber synapse variables call:

DEF_RWGHT_RANGE (module_number, var_number, var_minimum, var_maximum)

The arguments to these four routines are:

module_number: INTEGER variable

This is the number corresponding to the module whose bounds will be changed.

var_number: INTEGER variable

This is the number of the variable whose bounds will be changed. When changing cell and synapse variable bounds this can be 1, 2, or 3; when changing the input fiber variable bounds it can only be 1.

var_minimum: REAL variable

This is the minimum value of the variable that will be displayed. If the variable's value is less than this it will be displayed as if it equals this minimum value.

var_maximum: REAL variable

This is the maximum value of the variable that will be displayed. If the variable's value is greater than this it will be displayed as if it equals this maximum value.

F.3 Example

Assume that a network display has been created which consists of the following modules, and saved in the file named Mod3.Net.

Module 1 is of STANDARD type and has 4 cells, 4 input fibers, and has recurrent fibers. It will display the following variables:

DESIGNNET: NETWORK SIMULATION DISPLAY PACKAGE PAGE F-13
 Example

	Number	Minimum	Maximum
input fibers	1	0.0	1.0
cells	1	0.0	1.0
	2	-5.0	10.0
input synapses	1	-1.0	1.0
recurrent synapses	1	-3.0	3.0

Module 2 is of CORNER type connecting Module 1 and Module 2. It has 4 fibers and will display 1 variable (cell var 1).

	Number	Minimum	Maximum
fibers	1	0.0	1.0

Module 3 is of STANDARD type and has 4 inputs, 2 cells, and no recurrent fibers. It will display the following variables:

	Number	Minimum	Maximum
input fibers	1	0.0	1.0
cells	1	0.0	1.0
	2	-5.0	10.0
input synapses	1	-1.0	1.0

The following user's program calculates the values of all variables and calls the appropriate routines in DESIGNNET to display their values at each iteration.

```

program example
  real cellv1(4), cellv2(4), inp_var(4),mod1_iwts(4,4),
  & mod1_rwts(4,4), mod3_iwts(4,2)
  data mod1_iwts, mod1_rwts, mod3_iwts /16*0.5,
  & 16*3.0, 8*1.0/
C
C  INITIALIZE THE GRINNELL
C
  call gr_initialize (0, ' ')
  call gr_config444
C
C  PREPARE THE NETWORK
C
  call prepare_net ('MOD3.NET')
C
C  DISPLAY THE NETWORK SKELETON
C
  call disp_net_skel
C
C  DO THE FOLLOWING LOOP FOR 20 ITERATIONS (TIME STEPS)
C
  do 70 istep = 1,20

```

DESIGNNET: NETWORK SIMULATION DISPLAY PACKAGE PAGE F-14
Example

```
C      COMPUTATIONS FOR MODULE 1
C
C      do 10 numinp = 1,4
C          inp_var(numinp) = env(istep,numinp)
C              !(USER'S FUNCTION)
10      continue
C      do 40 numcell = 1,4
C          cellv1(numcell) = 0.0
C          do 20 numinp = 1,4
C              cellv1(numcell) = cellv1(numcell) +
C                  mod1_iwts(numinp,numcell) *
C                  inp_var(numinp)
20      continue
C      do 30 numrct = 1,4
C          cellv1(numcell) = cellv1(numcell) +
C              mod1_rwts(numrct,numcell) *
C              cellv1(numrct)
30      continue
C          cellv2(numcell) = cellv1(numcell) * istep / 4.0
40      continue
C
C      DISPLAY STATE OF MODULE 1
C
C      call update_mod_disp (1,cellv1,cellv2,dummy,
C          &           inp_var,mod1_rwts,dummy,dummy,mod1_iwts,
C          &           dummy,dummy,4,4)
C
C      DISPLAY STATE OF MODULE 2
C
C      call update_shortmod (2, cellv1)
C
C      COMPUTATIONS FOR MODULE 3
C
C      do 50 numcell = 1,2
C          inp_var(numinp) = cellv1(numinp)
50      continue
C      do 60 numcell = 1,2
C          cellv1(numcell) = 0.0
C          do 60 numinp = 1,4
C              cellv1(numcell) = cellv1(numcell) +
C                  mod3_iwts(numinp,numcell) *
C                  inp_var(numinp)
60      continue
C
C      DISPLAY STATE OF MODULE 3
C
C      call update_mod_disp (3,cellv1,dummy,dummy,
C          &           inp_var,dummy,dummy,dummy, mod3_iwts,
C          &           dummy,dummy,dummy, 2,4)
C
70      continue
    stop
```

end

F.4 Restrictions

The following limits are imposed by the DESIGNNET routines:

Quantity	Minimum	Maximum
Number of modules	1	15
Number of cells per module	1	10
Number of inputs per module	0	10
Number of shapes	0	40
Number of texts	0	60
Number of characters per text	1	70
Number of cell variables	0	3
Number of input fiber variables	1	1
Number of synapse variables	0	3

F.5 Error Messages

Error messages that appear at the terminal:

UNABLE TO OPEN filename

DESIGNNET was trying to open the file named 'filename' containing a stored network, but the file didn't exist. DESIGNNET will continue to ask for a valid file name until one is successfully opened.

PREMATURE END OF FILE filename

DESIGNNET was reading from the file named 'filename' containing a stored network. The end of the file was encountered before all data was read in. The file is in error, and the network should be recreated.

ERROR WHEN READING FILE filename

DESIGNNET was reading from the file named 'filename' containing a stored network. An invalid data type was encountered in the file. The network should be recreated.

*** VARIABLE MIN AND MAX THE SAME

A division by zero was encountered, because the minimum and the maximum values (the bounds) of a variable were set equal. The bounds can be changed by modifying the network using DESIGNNET.

*** VARIABLE THRESH AND MIN THE SAME

DESIGNNET: NETWORK SIMULATION DISPLAY PACKAGE PAGE F-16
Error Messages

*** VARIABLE THRESH AND MAX THE SAME

A division by zero was encountered, because the minimum or the maximum value of a variable was equal to the variable's threshold, which is the value at which the first synaptic variable's disk changes from red to green. The threshold is zero, so use DESIGNNET to change the minimum or maximum value (the bounds) of the variable so they are not zero.

NO BRAIN RETRIEVE PROGRAM LINKED

The main option UPDATE_NET was selected, but you did not link your BRAIN_RETRIEVE routine to DESIGNNET.

NO UPDATE NET DISP LINKED

The main option UPDATE_NET was selected, but you did not link your UPDATE_NET_DISP routine to DESIGNNET.

Error messages that appear on the Grinnell:

YOU ALREADY HAVE THE MAXIMUM NUMBER OF MODULES - COMMAND ABORTED

YOU ALREADY HAVE THE MAXIMUM NUMBER OF STRINGS - COMMAND ABORTED

NO DISPLAY ELEMENT NEAR THERE, TRY AGAIN

PLEASE PICK AGAIN - MODULES ONLY

WARNING: CURRENT STATE WILL BE LOST. REPEAT TO CONFIRM.

F.6 Program Origin

Authors: Rich Sutton and Chuck Anderson
Dept. of Computer and Information Sc.
University of Massachusetts
Amherst, MA 01002

Date: 1-Mar-1980

Funded by: Air Force Office of Scientific Research and the Avionics Laboratory (Air Force Wright Aeronautical Laboratories) through contract F33615-77-C-1191.

Source Language: FORTRAN IV-PLUS on VAX 11/780 VMS

Relevant Files: All in DR1:[ANWCA.DNET]
Designnet.doc

Designnet.exe
Dnetlib.olb
Netspecs.for
Netpoints.for
Desnet.for
Drawnet.for
Delink.com creates Desnet.Exe,
which is equivalent
to Designnet.Exe
Dir.doc explanation of files
in [ANWCA.DNET]

F.7 Summary of Method

To design or modify a network display:

RUN DR1:[ANWCA.DNET]DESIGNNET

When the network display is created, select the SAVE NET option to place the network display in a file. The name of this file must be used whenever the network display is wanted.

The following routines must be called by the user's program to display the network (see section 2.2):

GR_INITIALIZE
GR_CONFIG444
PREPARE_NET
DISP_NET_SKEL
UPDATE_MOD_DISP or UPDATE_SHORTMOD

To access these routines and the Grinnell graphics routines, the user's program must be linked by:

LINK user's_program,DR1:[ANWCA.DNET]DNETLIB/LIB,
[ANWRS.GRF]GLIB/LIB,[MOVIE]GRLIB1/LIB

F.8 Additional Documentation

The routines in the Grinnell graphics library are documented in the file: DR1:[MOVIE]GRDOC1.DOC .

The EXPER package, which simulates a robot and its environment, is documented in the file: DR1:[ANWCA.DNET]EXPER.DOC . This will explain the use of the data file named EXPER.BRA referred to in this document.

APPENDIX G

EXPERIMENTER: SIMULATION OF A ROBOT'S ENVIRONMENT

G.1 Description

EXPERIMENTER is a system for creating, modifying, manipulating, and simulating robots and their two-dimensional planar environments. The model world consists of an arbitrarily large number of independently moving robots, goal or landmark objects, and movement restricting barriers. A user provides the subroutines for the control of the robot, and this tool allows him to experiment with the capabilities of his control algorithm in a range of different environments much as a psychologist might investigate the behavior of some beast. The limitations on robot and environmental structures are particularly suited to the investigation of the learning, reasoning, and planning capabilities of a robot control algorithm.

EXPERIMENTER is a tool. EXPERIMENTER is not a complete system but only a starting place for an applications project. Only in the rarest of circumstances will major modifications and additions be unnecessary.

Nor is EXPERIMENTER everyman's robot simulation system. As a category of simulation objects, "robots and their environments" is far too broad to be handled by a unified system. Many different applications require the simulation of quite different world models. Thus, EXPERIMENTER is only appropriate for a restricted class of applications: It seems inappropriate for applications requiring a three dimensional model world or objects of complex internal structure. Where a two dimensional view of space is sufficient, as for spatial planning and reasoning tasks, then EXPERIMENTER may be very useful.

One way of thinking of EXPERIMENTER is as a software substitute for an actual robot. As a shape on a CRT screen

EXPERIMENTER: SIMULATION OF A ROBOTS' ENVIRONMENT PAGE G-2
Description

it is a poor substitute for a physical crawling beast, but for convenience and flexibility, it has the hardware version beat.

G.2 User Instructions

The use of EXPERIMENTER is described in three sections. First, the construction of a robot's brain routine is explained. The procedure for linking EXPERIMENTER with your routine and data files is then given. Finally, the execution of EXPERIMENTER is described by defining the functions of the options that are available during execution.

G.2.1 The Robot's Brain

The robots' control system, referred to as its brain, must be supplied by the user in the form of a FORTRAN subroutine named BRAIN. The BRAIN routine is passed an array of real values representing the state of the environment as perceived by the robots. BRAIN should process these values and generate an action to be performed by the robots. The action is represented by an array of real values passed back to EXPERIMENTER. A typical declaration for the subroutine would appear as:

```
SUBROUTINE BRAIN (ENVIR_ARRAY, ACTION_ARRAY)
```

If the state of the robots' brain will ever need to be saved and retrieved, then entry points BRAIN_SAVE and BRAIN_RETRIEVE should be included in BRAIN and would be declared as:

```
ENTRY BRAIN_SAVE (i/o unit number)  
ENTRY BRAIN_RETRIEVE (i/o unit number)
```

The following summary of the variable names used by EXPERIMENTER to represent the environment is included to aid in the design of a BRAIN routine. These variables can be accessed within the BRAIN routine by including the common block declarations contained in file DR1:[ANWRS.EXP]WORLDCOM.FOR.

AN INFORMAL DESCRIPTION OF THE WORLD MODEL OF EXPERIMENTER

As well as providing an "informal" description of the simulated system, the following constitutes a concise exposition of the data structures of experimenter. The common block structure should be apparent from the source code. In the following, I am using E as the "element of" symbol, and R (R_+) as a range to be the reals (positive reals).

A Discrete Time Model:

Components:

An infinite PLANE, upon which the following three kinds of objects are distributed.

NB BARRIERS, each of two ENDPOINTS

NL LANDMARKS, each a POINT OBJECT (PO)

NR ROBOTS, each a POINT OBJECT capable of motion

NR+NL SYMBOLS, the display representations of POINT OBJECTS

1 BRAIN, with

NA AFFERENTS (input lines)

NE EFFERENTS (output lines)

Descriptive Variables:

POX(PON)[t], POY(PON)[t] E R, PON=1,...,NR+NL

The x and y coordinate at time t of the PONth POINT OBJECT (PON is short for Point-Object-Number; POX is short for Point-Object-X-coordinate). The first NR POINT OBJECTS are ROBOTS, the last NL are LANDMARKS.

POSE(PON)[t] E R+, PON=1,...,NR+NL

The size (radius) of the PONth POINT OBJECT at time t.

BX(PN,BN)[t], BY(PN,BN)[t] E R, PN=1,2; BN=1,...,NB

The x and y coordinates at time t of the PNth ENDPOINT of the BNth BARRIER (BX is short for Barrier-X-coordinate; BN is short for Barrier-Number; PN is short for endPoint-NUmber.)

BW(BN)[t] E R+, BN=1,...,NB

The size (width) of the BNth BARRIER at time t.

AFF(AN)[t] E R, AN=1,...,NA

The signal on the ANth AFFERENT or input line to the BRAIN (AN is short for Afferent-Number).

EFF(EN)[t] E R, EN=1,...,NE

The signal on the ENth EFFERENT or output line from the BRAIN (EN is short for Efferent-Number). In the demo application, There are two EFFERENT lines

EXPERIMENTER: SIMULATION OF A ROBOTS' ENVIRONMENT PAGE G-4
User Instructions

for each robot in the world - so that EFF looks like DX1, DY1, DX2, DY2, ..., DXNR, DYNR, where (DXi, DYi) is the change in x and y coordinates being attempted this time step (DXi is short for Delta-X-of-the-i-th-robot).

Parameters (for display purposes):

SYMTYPE(SN) E {1,2,...,NST}, SN=1,...,NR+NL
The type of symbol (the shape) of the SNth SYMBOL.
In the demo application, each SYMBOL corresponds to the POINT OBJECT of the same index. (SYMTYPE is short for Symbol-Type; NST is short for Number-of-Symbol-Types)
SCR(SN), SCG(SN), SCB(SN) E {0,1,...,255},
SN=1,...,NR+NL:
The color of the SNth SYMBOL, corresponding in the demo application to the SNth POINT OBJECT. (SCR is short for Symbol-Color-Red.)
BCR(BN), BCG(BN), BCB(SN) E {0,1,...,255},
BN=1,...,NB:
The color of the BNth BARRIER (BCR is short for Barrier-Color-Red.)

Equations of Interaction:

POX(RN)[t+1] = POX(RN)[t] + EFF(RN*2-1) RN=1,...,NR
(in the demo application and if no barriers are hit; RN is short for Robot-Number.)
POY(RN)[t+1] = POY(RN)[t] + EFF(RN*2) RN=1,...,NR
EFF[t] = BRAIN[AFF[t]] AFF[t] = CALC_AFFERENTS[t] (in the demo application)

The standard way of using the network display package from EXPERIMENTER is to create an entry point called UPDATE_NET_DISP which takes no arguments and merely makes all necessary calls to UPDATE_MOD_DISP or UPDATE_SHORTMOD to update the display of your network. [All other network display functions are handled by EXPERIMENTER.] See the documentation for DESIGNNET for an explanation of the network display package.

G.2.2 Linking EXPERIMENTER

EXPERIMENTER must be linked with your BRAIN routine. This is done by executing the command file EXPLINK, which

EXPERIMENTER: SIMULATION OF A ROBOTS' ENVIRONMENT PAGE G-5
User Instructions

will ask you several questions regarding the following options:

Linking Options

GRADIENT DISPLAY

The gradient display shows the direction and magnitude of a robot's movements from points in the environment arranged in a rectangular array. Movement is shown by vectors originating at each of the points. Selecting this option also permits the display of the potentials of objects in the environment (see Edit Display Option POTENTIAL).

NETWORK DISPLAY

Use this option only if you have previously created a network display, using DESIGNNET, that corresponds to your BRAIN routine. This will link all the needed DESIGNNET routines to your program. Your BRAIN routine must include an UPDATE_NET_DISP entry point that performs the calls to UPDATE_MOD_DISP or UPDATE_SHORTMOD (see DESIGNNET documentation).

ROBOT TRACK DISPLAY

This option causes each robot to leave a trail as it moves in the environment. Each time a robot moves a line is drawn from its new location to its previous location.

NUMBERED SAVE/RETRIEVE FILES

This option allows you to have several different BRAIN states saved during execution. Each file is identified by an integer number which must be specified when saving and retrieving files. Without this option, the normal file name is used for the BRAIN state file.

The sequence of questions are shown below as they would appear at the terminal. The first line is what you type to execute EXPLINK.

```
@[anwrs.exp]explink your brain routine_file_name
GRADIENT DISPLAY OPTION (Y OR CR) ? y
NETWORK DISPLAY OPTION (Y OR CR) ? y
ROBOT TRACK DISPLAY OPTION (Y OR CR) ? y
NUMBERED SAVE/RETRIEVE FILES OPTION (Y OR CR) ? y
LINK EXIT
```

Data Files

There are 3 files that EXPERIMENTER can reference. They are described below with their names that EXPERIMENTER will initially refer to. When executing EXPERIMENTER you can select main option SET FILES to tell EXPERIMENTER the

EXPERIMENTER: SIMULATION OF A ROBOTS' ENVIRONMENT PAGE G-6
User Instructions

names of your files if they differ from the names below.

- | | |
|-----------|---|
| EXPER.ENV | Contains all information needed to display an environment. This can be retrieved to modify an existing environment or written into to save the current environment. |
| EXPER.BRA | Contains a state of the user's BRAIN routine. This can be retrieved to reset the BRAIN to a past state, or written into to save the current BRAIN state. Several different states can be saved in numbered files if you select the numbered files option when linking with EXPLINK. |
| EXPER.NET | Contains all information needed to display a network. This can be retrieved to display the network and subsequently updated by the BRAIN routine. EXPERIMENTER cannot change the network structure; use DESIGNNET to modify the network (see DESIGNNET documentation). |

If a file does not exist or there is some other error in opening it for reading, then no data will be read in, leaving the original configuration unchanged, and no error message will be given. If the numbered files option is used, then the file names will be 1.ENV, 1.BRA, 1.NET, 2.ENV, etc.

G.2.3 Executing EXPERIMENTER

To start executing EXPERIMENTER with your BRAIN routine type

RUN your_brain_routine_file_name

Since EXPERIMENTER is run in the DEBUG mode enter "go" in response to >DBG. Most of the interaction will be through the Grinnell. An option can be selected by moving cursor one into a box in the option list and pressing enter. Some options require subsequent option selections; to return to a previous option list press home, then enter.

Main Options

REFRESH

Redraw the current state of the environment or the network.

EDIT WORLD

EXPERIMENTER: SIMULATION OF A ROBOTS' ENVIRONMENT PAGE G-7
User Instructions

Allows you to change some aspect of the environment. Another list of options will appear and are described below, following the Main Options.

SIMULATE

Performs a number of time steps in the robot-environment interaction. One step is defined as one call to BRAIN and the execution of the resulting action. One of the following options must be selected to indicate the number of steps to perform and display. The accumulated number of steps will appear in the lower left corner of the screen.

SIMULATE Options

QUICK

Time steps will be continuously performed, but not displayed. Press enter to stop the simulation and to display the final state.

CONTINUOUS

Time steps will be continuously performed and the results of each step will be displayed. Press enter to stop the simulation.

SINGLE STEP

One time step is performed each time this option is selected, i.e., each time enter is pressed.

EDIT DISPLAY

Allows you to change the current display without affecting the structure of the environment. Another list of options will appear and are described below, following the EDIT WORLD options.

RETRIEVE ALL

All information in the environment, brain, and network data files is retrieved. This could change the current state of each, so a safety feature requires you to select this option a second time. Use the main option SET FILES to change the file name.

SAVE ALL

All information concerning the current states of the environment and the brain are saved in their respective data files. Use the main option SET FILES to change the file names.

PHOTO

The option list is erased and the display is refreshed. A photograph can then be taken of the display without the option list. Press enter to make the option list visible

EXPERIMENTER: SIMULATION OF A ROBOTS' ENVIRONMENT PAGE G-8
User Instructions

again.

MAKE GRADIENTS

Not functional at this time.

EXIT

Stops the execution of EXPERIMENTER. Since EXPERIMENTER is run in the DEBUG mode, you must enter "exit" in response to >DBG at the terminal.

EDIT WORLD Options

ADD ROBOT

This adds another robot to the environment (world). After selecting this option the following questions will be asked on the Grinnell to help you specify the new robot.

ADD ROBOT Questions

SYMBOL TYPE?

Several different symbols, e.g., square, circle, or box, will appear on the Grinnell. Place cursor one on the symbol that you want to represent the new robot and press enter. The symbol that you selected will be placed in the center of the screen.

SIZE?

Move cursor one towards or away from the screen center to reduce or enlarge the symbol. The cursor should be in track mode.

COLOR?

For this question you have two options.

COLOR Options

ARBITRARY

You select the color of the robot by choosing the red, green, and blue intensities by moving cursor one vertically and pressing enter for each color. Press home and enter when done.

BY MATCHING

The robot is given the color of another object in the environment by placing cursor one near the object having the desired color and pressing enter.

EXPERIMENTER: SIMULATION OF A ROBOTS' ENVIRONMENT PAGE G-9
User Instructions

POSITION?

With the cursor in track mode, move cursor one to position the robot in the environment. Press enter when done.

ADD LANDMARK

This adds a landmark to the environment. A landmark is a stationary element of the environment and has the capability of being observed by the robots. This is done by assigning a landmark a "scent", which is a function of the distance from the landmark. In this way, the robot can determine its location relative to the landmarks that it observes. The option MODIFY OBJECT under main option EDIT WORLD is used to assign a "scent" to an object. After selecting this option the following questions are asked on the Grinnell. Refer to the EDIT WORLD option ADD ROBOT for an explanation of these questions.

ADD LANDMARK Questions

SYMBOL TYPE?

SIZE?

COLOR?

POSITION?

ADD BARRIER

A barrier is a "wall" in the environment through which the robots cannot move. An object's "scent", however, will pass through a barrier. Barriers can be used to build mazes in which the robots can be placed. A barrier is specified by a rectangle that can be placed anywhere in the environment. Several questions will be asked on the Grinnell screen to help you specify the rectangle.

ADD BARRIER Questions

COLOR?

You must specify the color of the barrier in one of two ways, i.e., one of the following options must be selected. See the ADD ROBOT option for an explanation of each of these.

COLOR Options

ARBITRARY

BY MATCHING

1ST ENDPOINT?

An endpoint is the center point of one of the rectangle's short sides. Move cursor one to the position at which you want one end of the barrier. Press enter when done.

EXPERIMENTER: SIMULATION OF A ROBOTS' ENVIRONMENT PAGE G-10
User Instructions

2ND ENDPOINT?

In a similar manner, move cursor one to specify the second endpoint, and press enter.

WIDTH?

The width of the rectangle (barrier) is decreased or increased by moving cursor one towards or away from the line through the two endpoints. Cursor one should be in track mode. Press enter when done.

MODIFY OBJECT

This allows you to change any of the following aspects of an object in the environment. Place cursor one near the object you want to modify and press enter. The following options will appear.

MODIFY OBJECT Options

SYMBOL

You select a new symbol to represent the object by putting cursor one on the symbol and pressing enter.

COLOR

Choose a new color for the object in one of the following two ways. See ADD ROBOT option for an explanation of these options.

COLOR Options

ARBITRARY
BY MATCHING

AFF EFFECT

With this option you can change the way that this object influences the inputs to the robots' BRAIN routine. The object's influence can be referred to as its "scent". The object's influence on the inputs is determined by two items: the EFFECT and the POWER LAW. The EFFECT is set by selecting a real number for each input, or afferent. These numbers are the maximum values that the object can contribute to each input line. The POWER LAW determines the spatial extent of the object's influence. Its influence is maximal at the object and zero at its extent and beyond. Currently, its influence changes linearly from the object to the influence's extent. Each input value to the BRAIN routine is calculated by summing the contributions from all objects. Use the following options to change the object's influence.

EXPERIMENTER: SIMULATION OF A ROBOTS' ENVIRONMENT PAGE G-11
User Instructions

AFF EFFECT Options

SCALE

This allows you to change the scale in which the EFFECTS and the POWER LAW are displayed. Their values are not changed.

SCALE Options

DOUBLE EFFECT SCALE

Reduces the EFFECT scale.

HALF EFFECT SCALE

Expands the EFFECT scale.

DOUBLE POWER SCALE

Reduces the POWER LAW scale.

HALF POWER SCALE

Expands the POWER LAW scale.

ZERO ALL

Sets the EFFECTS and POWER LAWS for the object to zero.

POWER LAW

You select the extent of the object's influence by moving cursor one vertically while in track mode. Press enter when done.

EDIT EFFECT

You select the EFFECT of this object on each input value by moving cursor one vertically while in track mode. Press enter to change the next input value and when done.

NEXT ROBOT

Allows you to change the AFF EFFECT for the next robot.

SIZE-WIDTH

Alter the size of the object by moving cursor one away or towards the center of the object, or the line through the endpoints if the object is a barrier. The cursor should be in track mode. Press enter when done.

POSITION

Move the object to any point in the environment by moving cursor one to that point. The cursor should be in track mode. Press enter when done.

EDIT DISPLAY Options

EXPERIMENTER: SIMULATION OF A ROBOTS' ENVIRONMENT PAGE G-12
User Instructions

BACKGROUND

Specify the color of the background by choosing the red, green, and blue intensities by moving cursor one vertically and pressing enter for each color. Press home and enter when done.

GRADIENT

This option is valid only if you have compiled a gradient-calculating subroutine called BRAIN_NOSIDEEFF and have answered "y" to the linking option "GRADIENT DISPLAY?". If this has been done, then the gradient will be displayed. The gradient shows the direction and magnitude of the robot's movements from many points in the environment arranged in a rectangular array. The BRAIN_NOSIDEEFF subroutine should perform all of the BRAIN routine functions except changing the brain's state, i.e., just calculate the efferents given the afferents.

POTENTIAL

This option is valid only if you have answered "y" to the linking option "GRADIENT DISPLAY?". This will allow you to see the potential, or "scent", of an object in the environment. It is displayed as a color intensity surrounding the object. The object must be chosen by selecting its number in the option list that appears after selecting the POTENTIAL option. The objects' numbers correspond to the order in which they were created.

SHRINK

If the environment is currently displayed, then the environments dimensions are reduced. It appears as if you have stepped back from the environment to get a broader view. If the network is currently displayed, then the network's variable bounds are widened so the size or intensity of the variable displays are reduced.

EXPAND

This has the reverse effect of the SHRINK option.

TRANSLATE

The environment display can be shifted in any direction by placing cursor one on the point that you want to be shifted to the center of the screen. Press enter when you have positioned the cursor.

NETWORK

This option is valid only if you have previously created and saved a network with DESIGNNET (see DESIGNNET documentation) corresponding to the current BRAIN routine, and have included an UPDATE_NET DISP entry point that will display the values of the network variables.

EXPERIMENTER: SIMULATION OF A ROBOTS' ENVIRONMENT PAGE G-13
User Instructions

You also must have linked the network display option when performing EXPLINK. If this has been done, then the environment display is erased and the network is displayed. You can then select SIMULATE and observe the changes in the network display.

ENVIRONMENT

This option erases the network display and displays the current state of the environment. This option is the reverse of the NETWORK option.

SET RANGE

This option is valid only if the network is currently displayed. It allows you to change the range, or bounds, or the network display variables (see DESIGNNET documentation).

G.3 Example

The following FORTRAN IV-PLUS is an example of a BRAIN subroutine.

```
c-----  
c This BRAIN routine performs a Run and Twiddle algorithm.  
c If a robot perceived a stronger "scent" now than it did  
c one time step ago, it will continue in the same  
direction.  
c Otherwise, it selects a new direction at random.  
c-----  
c  
subroutine BRAIN (afferents, efferents)  
  
include 'DR1:[ANWRS.EXP]WORLDCOM.FOR'  
parameter maxnumrobots = 10  
real afferents(nr), efferents(nr*2),  
& oldlevel(maxnumrobots)  
integer i, unit, iseed  
  
if (nr .lt. 1) return !if no robots, then return  
  
do 20 i = 1, nr  
  if (afferents(i) - oldlevel(i) .le. 0.0) then  
    efferents(i*2-1) = 20.0 * (ran(iseed)**2 - 1)  
    efferents(i*2)    = 20.0 * (ran(iseed)**2 - 1)  
  endif  
  
  oldlevel(i) = afferents(i)
```

EXPERIMENTER: SIMULATION OF A ROBOTS' ENVIRONMENT PAGE G-14
Example

```
        return

c----- entry for BRAIN_SAVE -----
        entry BRAIN_SAVE (unit)
        write (unit,*) (oldlevel(i),i=1,nr)
        write (unit,*) iseed

        return

c----- entry for BRAIN_RETRIEVE -----
        entry BRAIN_RETRIEVE (unit)
        read (unit,*) (oldlevel(i),i=1,nr)
        read (unit,*) iseed

        return

c----- entry for calculating action gradient -----
        entry BRAIN_NOSIDEEFF (afferents, efferents)
c          This entry is meant to calculate the efferents as a
c          function of the afferents, but have no side effects
c          on the brain state. The resultant action choice is
c          interpreted as the direction the robot would tend to
c          move. This is used to display movement tendencies as
c          a function of position (see the Edit Display Option
c          GRADIENT).
        return

c----- entry for updating the network display -----
        entry UPDATE_NET_DISP
c          There is no network display for this example. If
c          there was, the appropriate calls to UPDATE_MOD_DISP
c          or UPDATE_SHORTMOD would go here.

        return

        end
```

EXPERIMENTER: SIMULATION OF A ROBOTS' ENVIRONMENT PAGE G-15
Example

G.4 Restrictions

The following restrictions are placed on the given quantities.

Quantity	Maximum
Number of robots	20
Number of point objects	60
Number of barriers	100
Number of input values passed to BRAIN	20
Number of output values returned by BRAIN	20

G.5 Error Messages (not applicable)

G.6 Program Origin

Author: Richard S. Sutton
Dept. of Computer and Information Sc.
University of Massachusetts
Amherst, MA 01002

Date: 1-Mar-1980

Funded by: Air Force Office of Scientific Research and the Avionics Laboratory
(Air Force Wright Aeronautical Laboratories) through contract F33615-77-C-1191.

Source Language: FORTRAN IV-PLUS on VAX 11/780 VMS

G.7 Summary of Method

Step 1.
Create a file containing a subroutine named BRAIN with entry points for BRAIN RETRIEVE and BRAIN SAVE, and optionally BRAIN_NOSIDEEFF and UPDATE_NET_DISP.

Step 2.
Compile the BRAIN subroutine.

EXPERIMENTER: SIMULATION OF A ROBOTS' ENVIRONMENT PAGE G-16
Summary of Method

Step 3.

Link EXPERIMENTER with your BRAIN routine by:

@[ANWRS.EXP]EXPLINK your_brain_routine_file_name

Step 4.

Run EXPERIMENTER. Create the environment and then simulate the robot-environment interaction.

G.8 Additional Documentation

The network display package, DESIGNNET, is described in the documentation file: DR1:[ANWCA.DNET]DESIGNNET.DOC

The graphics routines for the Grinnell display are documented in DR1:[MOVIE]GRDOC1.DOC.

BIBLIOGRAPHY

- Aizerman, M. A., Braverman, E. M., Rozonoer, L. I.: Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control* 25, 821-837 (1964)
- Albus, J. S.: Mechanisms of planning and problem solving in the brain. *Math. Biosci.* 45, 247-293 (1979)
- Alkon, D. L.: Voltage-dependent calcium and potassium ion conductances: A contingency mechanism for an associative learning model. *Science* 205, 810-816 (1979)
- Amari, S.: A mathematical approach to neural systems. In: *Systems neuroscience*. Metzler, J., ed. New York: Academic Press 1977
- Amari, S.: Neural theory of association and concept-formation. *Biol. Cybernetics* 26, 175-185 (1977)
- Amari, S., Arbib, M. A.: Competition and cooperation in neural nets. In: *Systems neuroscience*. Metzler, J., ed. New York: Academic Press 1977
- Anderson, J. A., Silverstein, J. W., Ritz, S. A., Jones, R. S.: Distinctive features, categorical perception, and probability learning: Some applications of a neural model. *Psychol. Rev.* 85, 413-451, (1977)
- Arbib, M. A.: *The metaphorical brain*. New York: Wiley-Interscience 1972
- Arbib, M. A.: Segmentation, schemas and cooperative computation. In: *Studies in mathematical biology*, part I: Cellular behavior and the development of pattern. Levin, S., ed. Math. Assoc. of America 1978

BIBLIOGRAPHY

PAGE 2

- Arbib, M. A.: A view of brain theory. To appear in:
Self-organizing systems, the emergence of order.
Yates, F. E., ed. New York: Plenum 1981a
- Arbib, M. A.: Perceptual structures and distributed motor control. To appear in: Handbook of physiology, vol. III: Motor control. Brooks, V. B., ed. Bethesda, MD: Amer. Physiol. Soc. 1981b
- Arbib, M. A., Kilmer, W. L., Spinelli, D. N.: Neural models and memory. In: Neural mechanisms and memory. Rosenzweig, M. R., Bennet, E. L., eds. Cambridge MA: The MIT Press 1976
- Ashby, W. R.: Design for a brain. New York: Wiley 1960
- Bain, A.: The Senses and the intellect (third edition). New York: Appleton 1874
- Barto, A. G., Sutton, R. S.: Prediction in classical conditioning: An adaptive element model. COINS Technical Report No. 80-02. Dept. of Computer and Information Science, University of Massachusetts, Amherst, MA, 1980.
- Barto, A. G., Sutton, R. S.: Landmark learning: An illustration of associative search. To appear in: Biol. Cybernetics (1981)
- Barto, A. G., Sutton, R. S., Brouwer, P.: Associative search network: A reinforcement learning associative memory. To appear in: Biol. Cybernetics (1981)
- Black, A. H., Dalrymple, A. J.: Reasoning in the rat reconsidered.
- Bledsoe, W. W., Browning, I.: Pattern recognition and reading by machine. Proc. Eastern Joint Comp. Conf., 225-232 (1959)
- Blodgett, H. C.: The effect of the introduction of reward upon the maze performance of rats. Univ. of Calif. Publ. Psychol. 4, 17-24 (1929)

Box, G., Jenkins, G.: Time series analysis: Forecasting and control. San Francisco: Holden Day 1976

Brindley, G. S.: Nerve net models of plausible size that perform many simple learning tasks. Proc. Roy. Soc. (Lond.) B 174, 173-191 (1969)

Brooks, R. S.: A balance principle for optimal access control. COINS technical report No. 80-20. Dept. of Computer and Information Science, University of Massachusetts, Amherst, MA, October, 1980

Bush, R. R., Mosteller, F.: Stochastic models for learning. New York: Wiley 1955

Busis, N. A., Weight, F. F., Smith, P. A.: Synaptic potentials in sympathetic ganglia: Are they mediated by cyclic nucleotides? Science 200, 1079-1081, (1973)

Campbell, D. T.: Blind variation and selective survival as a general strategy in knowledge-processes. In: Self-organizing systems, pp. 205-231. Yovits, M. C., Cameron, S., eds. New York: Pergamon 1962

Clark, W. A., Farley, B. G.: Generalization of pattern recognition in a self-organizing system. Proc. Western Joint Comp. Conf., 86-91 (1955)

Cooper, L. N.: A possible organization of animal memory and learning. In: Proceedings of the Nobel Symposium on collective properties of physical systems. Lundquist, B., Lundquist, S., eds. New York: Academic Press 1974

Craik, K. J. W.: The nature of explanation. Cambridge: Cambridge University Press 1934

Crane, H. D.: Beyond the seventh synapse: The neural marketplace of the mind. SRI Research Memorandum, Stanford Research Institute, Menlo park, CA, December 1978

BIBLIOGRAPHY

PAGE 4

Dawkins, R.: The selfish gene. New York: Oxford 1976

Dennett, D. C.: Why the law of effect will not go away.
In: Brainstorms. Montgomery, Vermont: Bradford 1973

Dickenson, A., Mackintosh, N. J.: Classical conditioning in
animals. Ann. Rev. Psychol. 29, 587-612 (1978)

Didday, R. L.: A model of visuomotor mechanisms in the frog
optic tectum. Math. Biosci. 30, 169-180 (1976)

Duda, R. O., Hart, P. E.: Pattern classification and scene
analysis. New York: Wiley 1973

Egger, M. D., Miller, N. E.: Secondary reinforcement in
rats as a function of informative value and reliability
of the stimulus. J. Exper. Psychol. 64, 97-104 (1962)

Eisenstein, B. A.: A self-learning estimator for tracking.
IEEE Trans. Syst., Man, Cybern. SMC-2, 2, 281-284
(1972)

Ellias, A. A., Grossberg, S.: Pattern formation, contrast
control, and oscillations in the short term memory of
shunting on-center-off-surround networks.
Biol. Cybernetics 20, 69-98 (1975)

Foster, C. C.: Content addressable parallel processors. New
York: Van Nostrand Reinhold 1976

Fraenkel, G. S., Gunn, D. L.: The orientation of animals:
Kineses, taxes and compass reactions. New York: Dover
1961

Freeman, W. J.: Mass action in the nervous system. New
York: Academic Press 1975

Frey, P. W., Sears, R. J.: Model of conditioning
incorporating the Rescorla-Wagner associative axiom, a
dynamic attention process, and a catastrophe rule.
Psychol. Rev. 85, 321-340 (1978)

BIBLIOGRAPHY

PAGE 5

Fukushima, K.: A model of associative memory in the brain.
Kybernetic 12, 58-53 (1973)

Galanter, E., Gerstenhaber, M.: On thought: The extrinsic theory. *Psychol. Rev.* 63, 218-227 (1956)

Greengard, P.: Possible role for cyclic nucleotides and phosphorylated membrane proteins in postsynaptic actions of neurotransmitters. *Nature* 260, 101-108 (1976)

Grossberg, S.: Some networks that can learn, remember, and reproduce any number of complicated space-time patterns. *J. Math. Mech.* 19, 53-91 (1969)

Grossberg, S.: Classical and instrumental learning by neural networks. In: *Progress in theoretical biology*. Rosen, R., Snell, F., eds. New York: Academic Press 1974

Grossberg, S.: Adaptive pattern classification and universal recoding: I. Parallel development and coding of neural feature detectors. *Biol. Cybernetics* 23, 121-134 (1976a)

Grossberg, S.: Adaptive pattern classification and universal recoding: II. Feedback, expectation, olfaction, illusions. *Biol. Cybernetics* 23, 187-202 (1976b)

Gregory, R. L.: On how so little information controls so much behavior. In: *Towards a theoretical biology, 2 sketches*. Waddington, C. H., ed. Edinburgh: Edinburgh University Press 1969

Grice, G. R.: The relation of secondary reinforcement to delayed reward in visual discrimination learning. *J. Exper. Psychol.* 38, 1-16 (1948)

Hanson, A. R., Riseman, E. M., eds.: *Computer vision systems*. New York: Academic Press 1978

BIBLIOGRAPHY

- Harth, E.: Visual perception: A dynamic theory.
Biol. Cybernetics 22, 169-130 (1976)
- Harth, E., Tzanakou E.: ALOPEX: A stochastic method for
determining visual receptive fields. Vision Res. 14,
1475-1482 (1974)
- Hawkins, J. K.: Self-organizing systems - a review and
commentary. Proc. IRE 49, 31-48 (1961)
- Hebb, D. O.: The organization of behavior. New York:
Wiley 1949
- Hilgard, E. R.: The nature of the conditioned response. I.
The case for, and against stimulus substitution.
Psychol. Rev. 43, 366-385 (1936)
- Hilgard, E. R., Bower, G. H.: Theories of learning (fourth
edition). Englewood Cliffs, New Jersey: Prentice-Hall
1975
- Ho, Y. C.: Differential games, dynamic optimization, and
generalized control theory. J. Optimization Theor. and
Applie. 6, 179-209 (1970)
- Holland, J. H.: Adaptation in natural and artificial
systems. Ann Arbor: University of Michigan Press 1975
- Hull, C. L.: Principles of behavior. New York:
Appleton-Century-Crofts 1943
- John, E. R., Schwartz, E. L.: The neurophysiology of
information processing and cognition.
Ann. Rev. Psychol. 29, 1-29 (1978)
- Kamin, L. J.: Predictability, surprise, attention and
conditioning. In: Punishment and aversive behavior.
Campbell, B. A., Church, R. M., eds. New York:
Appleton-Century-Crofts 1969

BIBLIOGRAPHY

PAGE 7

Kandel, E. R.: Cellular basis of behavior. San Francisco:
W. H. Freeman 1976

Kandel, E. R.: A cell-biological approach to learning.
Grass Lecture Monograph 1. Bethesda, MD: Society for
Neuroscience 1978

Kasyap R. L., Blaydon, C. C., Fu, K. S.: Stochastic
approximation. In: Adaptation, learning, and pattern
recognition systems: Theory and applications, pp.
339-354. Mendel, J. M., Fu, K. S., eds. New York:
Academic Press 1970

Kellogg, W. N.: Evidence for both stimulus-substitution and
original anticipatory responses in the conditioning of
dogs. *J. Exper. Psychol.* 22, 186-192 (1938)

Kesten, H.: Accelerated stochastic approximation.
Ann. Math. Statist. 29, 41-59 (1958)

Kilmer, W. L., McCulloch, W. S., Blum, J.: A model of the
vertebrate central command system. *Int. J. Man-Machine
Stud.* 1, 279-309 (1969)

Klopf, A. H., Gose, E.: An evolutionary pattern recognition
network. *IEEE Trans. Syst. Sci. Cybern.* SSC-5, 3,
247-250 (1969)

Klopf, A. H.: Brain function and adaptive systems - A
heterostatic theory. Air Force Cambridge Research
Laboratories Research Report AFCRL-72-0164, Bedford,
MA, 1972. (A summary appears in:
*Proc. Int. Conf. Syst., Man, Cybern., IEEE Syst., Man,
Cybern. Soc.*, Dallas, Texas, 1974)

Klopf, A. H.: Goal-seeking systems from goal-seeking
components: Implications for AI. *The Cognition and
Brain Theory Newsletter* 3, 2 (1979)

Klopf A. H.: The hedonistic neuron: A theory of memory,
learning and intelligence. Washington, D. C.:
Hemisphere Publishing Corp. 1981 (to be published)

- Kohonen, T., Oja, E.: Fast adaptive formation of orthogonalizing filters and associative memory in recurrent networks of neuron-like elements. *Biol. Cybernetics* 21, 85-95 (1976)
- Kohonen, T.: Associative memory: A system theoretic approach. Berlin: Springer 1977
- Koshland, D. E. Jr.: A model regulatory system: Bacterial chemotaxis. *Physiol. Rev.* 59, 811-862 (1979)
- Kuipers, B. J.: Representing knowledge of large-scale space. M.I.T. Artificial Intelligence Laboratory Report AI-TR-418. Cambridge, MA, 1977
- Lashley, K. S.: The problem of serial order in behavior. In: Cerebral mechanisms of behavior: The Hixon Symposium. Jeffress, L. P., ed. New York: Wiley 1951
- Libet, B., Kobayashi, H., Tanaka, T.: Synaptic coupling into the production and storage of a neuronal memory trace. *Nature* 258, 155-157 (1975)
- Longuet-Higgins, H. C.: Holographic model of temporal recall. *Nature* 217, 104 (1968a)
- Longuet-Higgins, H. C.: The non-local storage of temporal information. *Proc. Roy. Soc. (Lond.) B* 171, 327 (1968b)
- Longuet-Higgins, H. C., Willshaw, D. J., Buneman, O. P.: Theories of associative recall. *Rev. Biophys.* 3, 223-244 (1970)
- Luce, R. D., Raiffa, H.: Games and decisions. New York: Wiley (1957)
- Lund, R. D.: Development and plasticity of the brain. New York: Oxford 1973

- Mach, E.: On the part played by accident in invention and discovery. *Monist* 6, 161-175 (1896)
- MacKay, D. M.: The epistemological problem for automata. In: *Automata studies*, pp. 235-251. Shannon, C. E., McCarthy, J., eds. Princeton, N.J.: Princeton University Press 1955
- Mackintosh, N. J.: *The psychology of animal learning*. New York: Academic Press 1974
- Marr, D.: A theory of cerebellar cortex. *J. Physiol.* 202, 437-470 (1969)
- Marshak, J., Radner R.: *Economic theory of teams*. New Haven: Yale University Press 1972
- Mandel, J. M.: Synthesis of quasi-optimal switching surfaces by means of training techniques. In: *Adaptation, learning, and pattern recognition systems: Theory and applications*, pp. 163-195. Mendel, J. M., Fu, K. S., eds. New York: Academic Press 1970
- Mendel, J. M., Fu, K. S., eds.: *Adaptive, learning, and pattern recognition systems: Theory and applications*. New York: Academic Press 1970
- Mendel, J. M., McLaren, R. W.: Reinforcement-learning control and pattern recognition systems. In: *Adaptive, learning, and pattern recognition systems: Theory and applications*, pp. 287-317. Mendel, J. M., Fu, K. S., eds. New York: Academic Press 1970
- Michie, D., Chambers, R. A.: BOXES: An experiment in adaptive control. *Machine Intelligence* 2, pp. 137-152. Dale E., Michie, D., eds. Edinburgh: Oliver and Boyd 1968
- Miller, G. A., Galanter, E., Pribram, K. H.: *Plans and the structure of behavior*. New York: Henry Holt and Co. 1960

Milner, J. S., Kashland, D. E., Jr.: Sensory electrophysiology of bacteria: Relationship of the membrane potential to motility and chemotaxis in Bacillus subtilis. Proc. Natl. Acad. Sci. USA 74, 4752-4755 (1977)

Milner, P. M.: The cell-assembly: Mark II. Psychol. Rev. 64, 242-252 (1957)

Minsky, M. L.: Steps toward artificial intelligence. Proc. IRE 49, 3-30 (1961)

Minsky, M. L., Papert, S.: Perceptrons: An introduction to computational geometry. Cambridge, MA: MIT Press 1969

Minsky, M. L., Papert, S.: Artificial intelligence progress report. MIT Artificial Intelligence Laboratory, Memo 252, January 1, 1972

Minsky, M. L., Selfridge, O. G.: Learning in random nets. In: Information theory: Fourth London Symposium. Cherry, C., ed. London: Butterworths 1951

Moore, J. W.: Brain processes and conditioning. In: Mechanisms of learning and motivation: A memorial volume to Jerzy Konorski. Dickenson, A., Boakes, R. A., eds. Hillsdale, New Jersey: Erlbaum 1979

Nakano, K.: Associatron - a model of associative memory. IEEE Trans. Syst., Man, Cybern. SMC-2, 3, 380-388 (1972)

Narendra, K. S., Thatachar, M. A. L.: Learning automata - a survey. IEEE Trans. Syst., Man, Cybern. SMC-4, 4, 323-334 (1974)

Nathanson, J. A.: Cyclic nucleotides and nervous system function. Physiol. Rev. 57, 157-256 (1977)

Nilsson, N. J.: Learning Machines. New York: McGraw-Hill 1955

Nilsson, N. J.: Artificial intelligence. Artificial Intelligence Center Technical Note 89, Stanford Research Institute, Menlo Park, CA, 1974. (Also presented at IFIP Congress 1974, Stockholm, Sweden)

Palm G.: On associative memory. Biol. Cybernetics 36, 19-31 (1980)

Perel'man, I. I.: Method of self-adjustment of step search systems. Translated from Avtomatika i Telemekhanika 4, 80-93 (April, 1967)

Piaget, J.: The construction of reality in the child. New York: Basic Books 1954

Poggio, T.: On optimal nonlinear associative recall. Biol. Cybernetics 19, 201-209 (1975)

Poincaré, H.: L'invention mathématique. Bull. Inst. Gen. Psychol. 8, 175-187 (1908)

Poincaré, H.: Mathematical creation. In: The foundations of science. Poincaré, H., ed. New York: Science Press 1913

Powers, W. T.: Behavior: The control of perception. Aldine 1973

Prokasy, W. F., Gormezano, I.: The effect of US omission in classical aversive and appetitive conditioning of rabbits. Animal Learning and Behavior 7, 80-88 (1979)

Rasmussen, H., Jensen, P., Lake, W., Friedman, N., Goodman, D. B. P.: In: Advances in cyclic nucleotide research, vol. 5. Drummond, G. I., Greengard, P., Robison, G. A., eds. New York: Raven Press 1975

Rescorla, R. A., Solomon, R. L.: Two-process learning theory: Relationships between Pavlovian conditioning and instrumental learning. Psychol. Rev. 74, 151-182 (1967)

Rescorla, R. A., Wagner, A. R.: A theory of Pavlovian conditioning: Variations in the effectiveness of reinforcement and non-reinforcement. In: Classical conditioning II: Current research and theory. Black, A. H., Prokasy, W. F., eds. New York: Appleton-Century-Crofts 1972

Rochester, N., Holland, J. H., Haibt, L. H., Duda, W.: Tests on a cell assembly theory of action of the brain, using a large digital computer. IRE Trans. Infor. Theor. 2, 80-93 (1956)

Rosenblatt, F.: Principles of neurodynamics. New York: Spartan Books 1962

Samuel, A. L.: Some studies in machine learning using the game of checkers. IBM J. Res. and Dev. 3, 210-229 (1959)

Saridis, G. S.: Learning applied to successive approximation algorithms. IEEE Trans. Syst. Sci. Cybern. SMC-6, 2, 97-103 (1970)

Schneiderman, N.: Interstimulus interval function of the nictitating membrane response of the rabbit under delay versus trace conditioning. J. Comp. Physiol. Psychol. 62, 397-402 (1966)

Schneiderman, N., Gormezano, I.: Conditioning of the nictitating membrane of the rabbit as a function of the CS-US interval. J. Comp. Physiol. Psychol. 57, 188-195 (1964)

Selfridge, O. G.: Pandemonium: A paradigm for learning. In: Proceedings of the Symposium on Mechanization of Thought Processes. Blake, D. V., Uttley, A. M., eds. London: H. M. Stationery Office 1959

Selfridge, O. G.: Tracking and trailing: Adaptation in movement strategies. Unpublished draft, August 1, 1978

Simon, H. A.: The sciences of the artificial. Cambridge, Mass.: The MIT Press 1969

BIBLIOGRAPHY

Smith, J. Maynard: The evolution of sex. Cambridge: Cambridge University Press 1978

Smith, M. C., Coleman, S. R., Gormezano, I.: Classical conditioning of the rabbit's nictitating membrane response at backward, simultaneous and forward CS-US intervals. *J. Comp. Physiol. Psychol.* 69, 226-231 (1969)

Sommerhoff, G.: Logic of the living brain. New York: Wiley 1975

Spinelli, D. N.: OCCAM: A computer model for a content addressable memory in the central nervous system. In: The biology of memory. Pribram, K., Broadbent, D., eds. New York: Academic Press 1970

Stent, G. S.: A physiological mechanism for Hebb's postulate of learning. *Proc. Nat. Acad. Sci. USA* 70, 997-1001 (1973)

Sutton, R. S.: A formalization of Klopf's heterostatic neuron. Unpublished report 1977

Sutton, R. S.: Single channel theory: A neuronal theory of learning. *Brain Theory Newsletter* 3, 72-75 (1978a)

Sutton, R. S.: Learning theory support for a single channel theory of the brain. Unpublished report 1978b

Sutton, R. S.: A unified theory of expectation in classical and instrumental conditioning. Stanford undergraduate thesis 1978c

Sutton, R. S., Barto, A. G.: Toward a modern theory of adaptive networks: Expectation and prediction. *Psychol. Rev.* 88 (1981)

Thorndike, E. L.: Animal intelligence. New York: Macmillan 1911

Thorpe, W. H.: Learning and instinct in animals. London: Methuen and Co. Ltd. 1956

Tolman, E. C., Honzik, C. H.: "Insight" in rats. Univ. Calif. Publ. Psychol. 4, 215-232 (1930)

Tsetlin, M. L.: Automaton theory and modeling of biological systems. New York: Academic Press 1973

Tsyplkin, Y. Z.: Adaptation and learning in automatic systems. Translated by Z. J. Nikolic. New York: Academic Press 1971

Tzanakou, E., Michalak, R., Harth, E.: The alopex process: Visual receptive fields by response feedback. In preparation.

Uhr, L., Vossler, C.: A pattern recognition program that generates, evaluates and adjusts its own operators. Proc. Western Joint Comp. Conf., 555-569 (1961)

Uttley, A. M.: The informon: A network for adaptive pattern recognition. J. theor. Biol. 27, 31-57 (1970)

Uttley, A. M.: The informon in classical conditioning. J. Theo. Biol. 49, 355-376 (1975)

Uttley, A. M.: A two-pathway informon theory of conditioning and adaptive pattern recognition. Brain Res. 102, 23-35 (1976a)

Uttley, A. M.: Simulation studies of learning in an informon network. Brain Res. 103, 37-53 (1976b)

Uttley, A. M.: Neurophysiological predictions of a two-pathway informon theory of neural conditioning. Brain Res. 102, 55-70 (1976c)

Uttley, A. M.: Information transmission in the nervous system. London: Academic Press 1979

von Baumgarten, F. J.: Plasticity in the nervous system at the unitary level. In: The neurosciences second study program. Schmitt, F. O., ed. New York: Rockefeller University Press 1970

von der Malsburg, C.: Self-organization of orientation sensitive cells in the striate cortex. Kybernetic 14, 85-100 (1973)

von Neumann, J., Morgenstern, O.: Theory of games and economic behavior. Princeton N. J.: Princeton University Press 1953

Weight, F. F., Erulkar, S. D.: Modulation of synaptic transmitter release by repetitive postsynaptic action potentials. Science 193, 1023-1025 (1976)

Weight, F. F., Schulman, J. A., Smith, P. A., Busis, N. A.: Long-lasting synaptic potentials and the modulation of synaptic transmission. Fed. Proc. 38, 2094-2094 (1979)

Widrow, B., Narendra, K. G., Maitra, S.: Punish/Reward: Learning with a critic in adaptive threshold systems. IEEE Trans. Syst., Man, Cybern. SMC-3, 5, 455-465 (1973)

Widrow, G., Hoff, M. E.: Adaptive switching circuits. In: 1960 IRE WESCON Convention Record, Part 4, 96-104 (1960)

Wigström, H.: A neuron model with learning capability and its relation to mechanisms of association. Kybernetic 12, 204-215 (1973)

Willshaw, D. J., Buneman, O. P., Longuet-Higgins, H. S.: Non-holographic associative memory. Nature 222, 960-962 (1969)

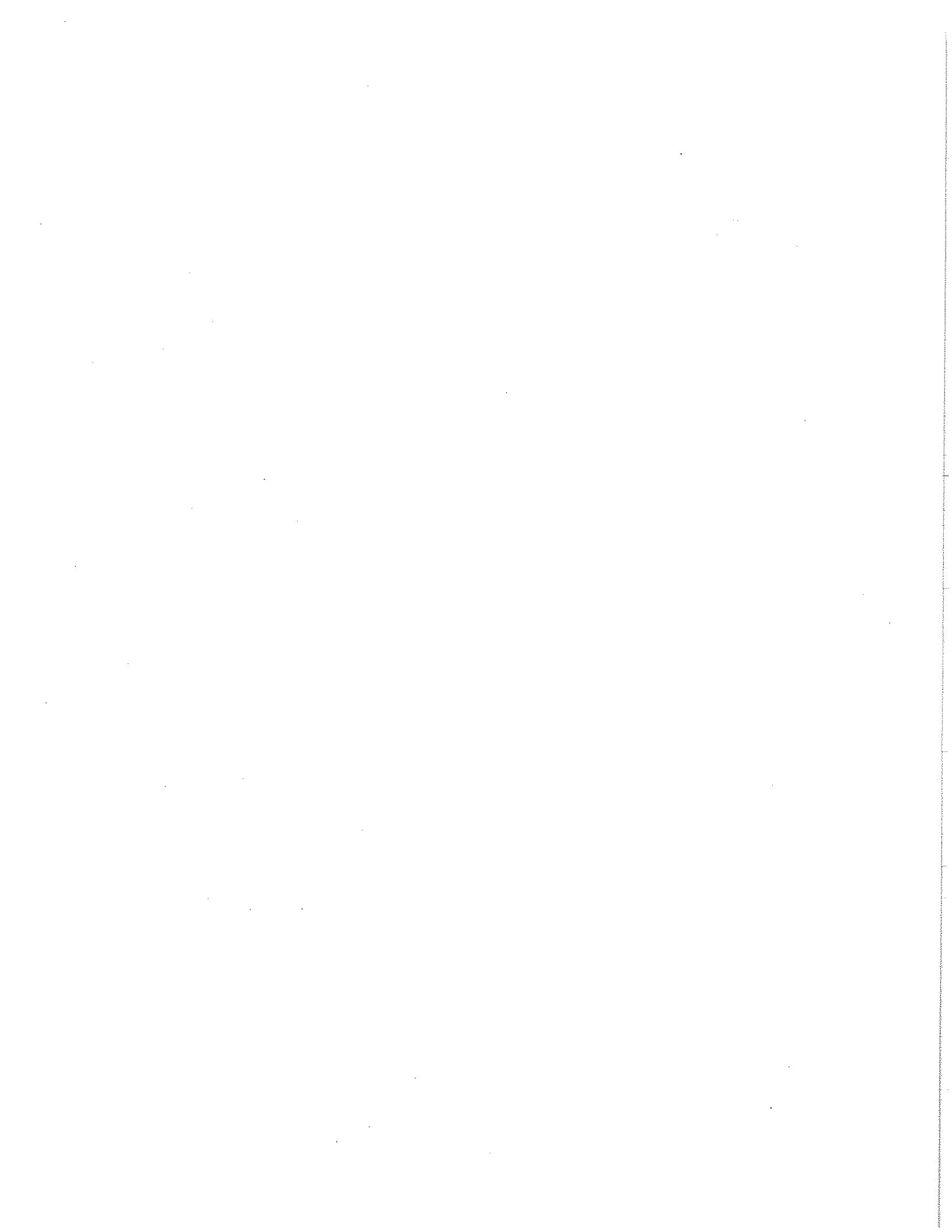
Wood, C. C.: Variations on a theme by Lashley: Lesion experiments on the neural model of Anderson, Silverstein, Ritz, and Jones. Psychol. Rev. 85, 582-591 (1978)

Woody, C. D.: If cyclic GMP is a neuronal second messenger what is the message? In: Cholinergic mechanisms and psychopharmacology. Jenden, D. J., ed. New York: Plenum 1976

Woody, C. D., Carpenter, D. O., Gruen, E., Knispel, J. D., Crow, T. J., Black-Cleworth, P.: Prolonged increases in resistance of neurons in cat motor cortex following extracellular iontophoretic application of acetylcholine (ACh) and intracellular current injection. Fed. Amer. Soc. Exp. Biol. 33, 399 (1974)

Yemini, Y., Kleinrock: Access control or, silence is golden . . . DSN-Distributed Sensor Networks, ISI Working Paper 12, USC Information Sciences Institute, Marina del Rey, CA, 1978

Zimmer-Hart, C. L., Rescorla, R. A.: Extinction of Pavlovian conditioned inhibition. J. Comp. Physiol. Psychol. 86, 837-845 (1974)



ALL SALES ARE FINAL

**NTIS strives to provide quality products, reliable service, and fast delivery.
Please contact us for a replacement within 30 days if the item you receive
is defective or if we have made an error in filling your order.**

► E-mail: customerservice@ntis.gov
► Phone: 1-888-584-8332 or (703)605-6050

Reproduced by NTIS

National Technical Information Service
Springfield, VA 22161

*This report was printed specifically for your order
from nearly 3 million titles available in our collection.*

For economy and efficiency, NTIS does not maintain stock of its vast collection of technical reports. Rather, most documents are custom reproduced for each order. Documents that are not in electronic format are reproduced from master archival copies and are the best possible reproductions available.

If you have questions concerning this document or any order you have placed with NTIS, please call our Customer Service Department at 1-888-584-8332 or (703) 605-6050.

About NTIS

NTIS collects scientific, technical, engineering, and related business information – then organizes, maintains, and disseminates that information in a variety of formats – including electronic download, online access, DVD, CD-ROM, magnetic tape, diskette, multimedia, microfiche and paper.

The NTIS collection of nearly 3 million titles includes reports describing research conducted or sponsored by federal agencies and their contractors; statistical and business information; U.S. military publications; multimedia training products; computer software and electronic databases developed by federal agencies; and technical reports prepared by research organizations worldwide.

For more information about NTIS, visit our Web site
at <http://www.ntis.gov>.



**Ensuring Permanent, Easy Access to
U.S. Government Information Assets**



U.S. DEPARTMENT OF COMMERCE
Technology Administration
National Technical Information Service
Springfield, VA 22161 (703) 605-6000