A THEORY OF NEUROMIME NETS CONTAINING RECURRENT INHIBITION, WITH AN ANALYSIS OF A HIPPOCAMPUS MODEL

Thesis for the Degree of Ph. D.

MICHIGAN STATE UNIVERSITY

DUANE G. LEET

1971

This is to certify that the

thesis entitled

A THEORY OF NEUROMIME NETS CONTAINING RECURRENT INHIBITION, WITH AN ANALYSIS OF A HIPPOCAMPUS MODEL.

presented by

Duane G. Leet

has been accepted towards fulfillment of the requirements for

Ph. D. degree in Systems Science

[illegible]
Major professor

Date April 30, 1971

O-7639

realize
intercor
Some fu

CA3 sec
contains
logic ur
logic ur
functior
connect
there e
some el

transfo
sequenc
with the
of outpu
derived
particu
possibl

its func
trainer
the trai
system
are dis

ABSTRACT

A THEORY OF NEUROMIME NETS CONTAINING RECURRENT INHIBITION, WITH AN ANALYSIS OF A HIPPOCAMPUS MODEL

By

Duane G. Leet

A novel system component called the functal can be set to realize any one of many different functions. A functal net is an interconnected array of functals, function generators, and delays. Some fundamental time-domain properties of these nets are developed.

A functal net model of recurrent inhibition as found in the CA3 sector of mammalian hippocampus is presented. The model contains a rank of functals, which are somewhat like adaptive threshold logic units, and a rank of function generators, which are threshold logic units. The two ranks are interconnected through delays, and the function generators inhibit the functals. The only assumption on the connectivity between ranks is that, for each element in the first rank, there exists at least one direct circuit path from that element through some element of the second rank and then back to itself.

The most important characteristic of the model's input-output transformation is that a single input can be transformed into a sequence of outputs. This sequence terminates, for a given input, with the continuous repetition of either a single output or a sequence of outputs. Some properties of the model's output sequences are derived, and an algorithm is developed for generating, for any particular N-functal net, all output sequences which that net could possibly produce.

A trainable functal net is one in which the functions realized by its functals are under the control of an external structure called the trainer, which operates according to a specified algorithm. Both the trainer and the trainable functal net are part of a new canonical system called a functal system, some fundamental properties of which are discussed.

Th
model of
certain c
training
realized
function
as long
fication
T
is place
selecti
derived

Duane G. Leet

The CA3 model is incorporated into an automata theoretic model of the hippocampus that is designed to take advantage of certain of the CA3 model's properties. There does not exist a training algorithm for this model that can always change the function realized by one of its functals to any other arbitrarily specified function. But an algorithm is given that can produce defined changes as long as the parameters of the CA3 model meet certain specifications.

The function realized by a functal system model whenever it is placed in a new environment is called the initial function. The selection of initial functions is discussed, and an algorithm is derived to select them automatically.

A THEORY OF NEUROMIME NETS CONTAINING RECURRENT INHIBITION, WITH AN ANALYSIS OF A HIPPOCAMPUS MODEL

By

Duane G.^{ary} Leet

A THESIS

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Electrical Engineering and Systems Science

1971

me
onl
oth

anc
me
tua
enc
enc

the
of

ACKNOWLEDGEMENTS

The author gratefully acknowledges the aid and encouragement given by his academic advisor, Dr. William Kilmer, not only on matters related to the research project, but on enumerable other matters as well.

The author's father, Gerald Leet, his grandparents, James and Ruby Leet, his inlaws, Floyd and Irene Layman, and other members of his wife's family have contributed financial and spiritual support during his graduate studies. The author is also deeply endebted to the late Dr. Leroy Augenstein for his support and encouragement.

Finally, this thesis could not have been completed without the inexhaustible patience, cheerfulness, and sustaining support of the author's wife, Chris.

HW(X

01x0x

f(01x

$X \subset$

(X c

iff

s. t.

dis

$\mathcal{P}_n$

$\mathcal{T}_n$

$\mathcal{B}$

$|C$

$\| V$

X

n

NOTATION

HW(X)	Hamming weight: HW(0101101) = 4
01x0x	$\{01000, 01001, 01100, 01101\}$
f(01x0x) = 1	f(01000) = f(01001) = f(01100) = f(01101) = 1
$X \subset Y$ (X covers Y)	$HW(X) \geq HW(Y)$ and $X \cdot Y = \|Y\|^2$
iff	if and only if
s. t.	such that
disc. fcn.	discriminant function
$\mathcal{P}_n$	$\{V : V = (v_1, v_2, \ldots, v_n)$ and v_i a non-negative integer, $1 \leq i \leq n\}$
$\mathcal{T}_m$	$\{V = (v_1, v_2, \ldots, v_m) : v_i \epsilon \{0, 1, 2\}\}$
$\mathcal{B}_n$	$\{V = (v_1, v_2, \ldots, v_n) : v_i \epsilon \{0, 1\}\}$
$\|\mathcal{A}\|$	the cardinality, or number of elements, of the set $\mathcal{A}$
$\|Y\|^2$	$\Sigma_{i=1}^n y_i^2$, $Y = (y_1, \ldots, y_n)$
$X \cdot Y$	$\Sigma_{i=1}^n x_i y_i$, Y as above and X of the same form
mf	mossy fiber

TABLE OF CONTENTS

Table

Table

Tabl

Tabl

LIST OF TABLES

LIST OF FIGURES

1.1.

and
whic
ceiv
neur
neur
rec
the
inhi
imp
con

ner
hip
rea
sci

he
Wi
str
lat

CHAPTER 1

INTRODUCTION

1.1. What is the Function of Recurrent Inhibition?

Recurrent inhibition can be described in terms of components and connectivity and interneuronal relationships. The components, which are neurons, are arranged in two ranks. The first rank receives inputs from elsewhere in the nervous system and from neurons in the second rank; it sends outputs elsewhere and to neurons in the second rank (Figure 1.1.1). The second rank receives inputs only from the first rank and sends outputs only to the first rank. The interneuronal relationship, termed interneuronal inhibition, holds when a neuron in the second rank decrements the impulse frequencies of those neurons in the first rank to which it is connected.

Recurrent inhibition is found in many regions of vertebrate nervous systems: sensory systems [1], the cerebellum [2], the hippocampus [3], and perhaps the spinal cord [4, 5]. For this reason, understanding its function should be of interest to neuroscientists.

When a neuroscientist speaks of the function of a structure, he is usually referring to its specialized actions or purposes. Within this context, a number of functions have been attributed to structures containing recurrent inhibition, or its close relative, lateral inhibition:

1. enhancement of contrast [1] and the detection of edges [6],
2. blockage of low-level inputs [1],
3. amplification of time-varying signals of certain frequencies [7],

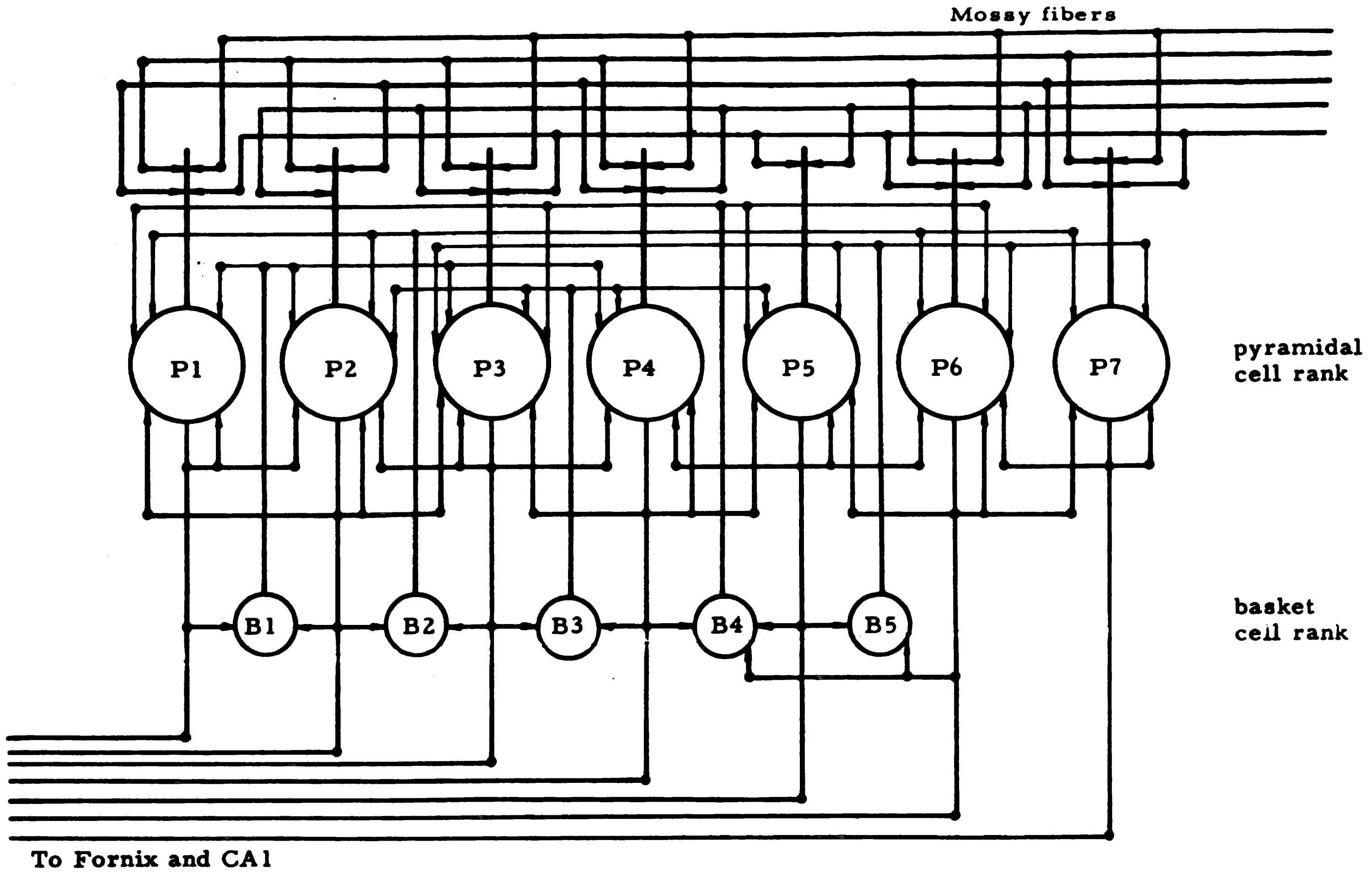


Figure 1. 1. 1. A schematic of a section of the CA3 sector of the hippocampus.

A schematic of a section of the CA3 sector of the hippocampus, adapted from Kilmer [12]. The mossy fibers from the fascia dentata come in at the top of the picture and wind their way among the apical dendrites of the pyramidal cells P1-P7. The inputs from the septum impinge on both the basal and apical dendrites of the pyramidal cells. They have not been shown in the schematic, however, since connectivity rules have not been defined. Two other inputs to the pyramids are not shown, the perforant and commissural fibers. The reasons for their ommission in this schematic are discussed by Kilmer. The axons of the pyramids are the outputs of the sector; they will eventually bifurcate, one branch heading toward the septum and the other finding its way to the CA1-CA2 sector. These axons also have collaterals to the basal and apical dendrites of other pyramids and to the somata and dendrites of the basket cells B1-B5. The basket cell axons feed back to the somata of the pyramids, completing the recurrent inhibition loop. The connectivity guidelines are believed to be: any one mossy fiber feeds a large number of pyramids, perhaps skipping many pyramids for each one it relates to. The basket cell field is smaller and more compact, and the pyramid field for both the excitatory pyramid to pyramid connections and the pyramid to basket connections is very small, on the order of a few cells. The connectivities shown in the schematic are exemplary.

st
it
ar
pa
"
e
p

f
n
i
w
a
i

I

t
s
I
w
a
t

c

4. selective response to signal patterns flowing in one direction in a two-dimensional space [8],
5. the generation of two periodic signals approximately 180 degrees out of phase with each other from a single input [9],
6. production of quasi-impulse responses to step inputs [10], and
7. preferential response to stimuli having certain orientations [11].

Mathematical readers usually interpret the function of a structure to be the list of input-output correspondences produced by it, where the word "list" presupposes only an algorithm (essentially an ordered set of instructions) that can generate any input-output pair of the list. With this interpretation, the neuroscientist's "functions" can be regarded as a list of vaguely defined algorithms, each of which indicates how a certain subset of the set of all possible inputs is related to the set of outputs.

In order to avoid confusion over these two meanings of function, the following convention will be adopted: if "function" is meant in the neurophysiological sense, the word "task" will be used in its place; if "function" is meant in the mathematical sense, the word "function" will be used. This thesis represents the first attempt known to the author to investigate the <u>function</u> of recurrent inhibition.

1. 2. Hippocampus Morphology

In general, in order to determine the function of any operational unit, it is necessary to measure its inputs and outputs simultaneously. The vertebrate central nervous system does not lend itself to this approach because the inputs and outputs of its various subunits are for the most part inaccessable, undecipherable, and apparently highly variable in the frequency domain. Furthermore, the subunits themselves change rapidly with time.

An alternative approach to the determination of the function of the operational unit is to model it mathematically or by computer

sim
this
phy
App
The
sec
tion
tur
ori
exi
de
hip
is t
mo

1. 3

the
fou

the
ne
ca
a r
pe
1.
tir
af
an
re
an
tu
At
as
mo
as

simulation (or some blend of both). The hippocampus is well suited to this approach. In particular, a wealth of both morphological and neurophysiological data exists on it (see Kilmer [12], References and Appendix B), which makes component modeling comparatively easy. The hippocampus also has a highly stylized connectivity and the CA3 sector clearly exhibits all of the known indicators of recurrent inhibition (see Figure 1.1.1); thus its circuit organization is easily caricatured. Two kinds of inputs (ignoring the commissural fibers) and their origins, plus two kinds of outputs and their destinations are known to exist (see Figure 1.2.1). Thus, the inputs and outputs of any model are defined and their characteristics can be compared with the available hippocampal electrophysiological data. In summation, the hippocampus is the neural structure of choice for an investigation by mathematical model and computer simulation of the function of recurrent inhibition.

1. 3. The Expositional Problem

The following example points up the difficulty of communicating the principles of circuit actions for a neural net of the complexity found in Figure 1. 1. 1 and of concisely describing the net's function.

Consider the neuron net shown in Figure 1. 1. 1, ignoring all of the direct pyramid-to-pyramid connections. Assume all activity in the net is allowed to die out, and then apply an input to the net sufficient to cause P3 to produce a moderate number of pulses per second (fire at a moderate rate) and to cause P5 to produce a large number of pulses per second (fire at a high rate). If this occurs at time t_o (Figure 1. 3. 1), and the leading edges of both trains of pulses require the same time to reach the basket cell rank, then both B3 and B4 will be affected at time t_1; suppose B3 responds by firing at a moderate rate and B4 responds by firing at a high rate. Assuming these pulse trains require the same time to travel to the pyramidal cell rank, both P3 and P5 will be affected at the time t_2; suppose P3 reacts by completely turning off and P5 reacts by decreasing its output to a moderate rate. At some later time t_3 these changes will be felt by the basket cells; as a result suppose B3 turns off and B4 decreases its output to a moderate rate. At time t_4 these changes will be felt by the pyramids; as a result suppose P3 begins firing at a moderate rate again and P5

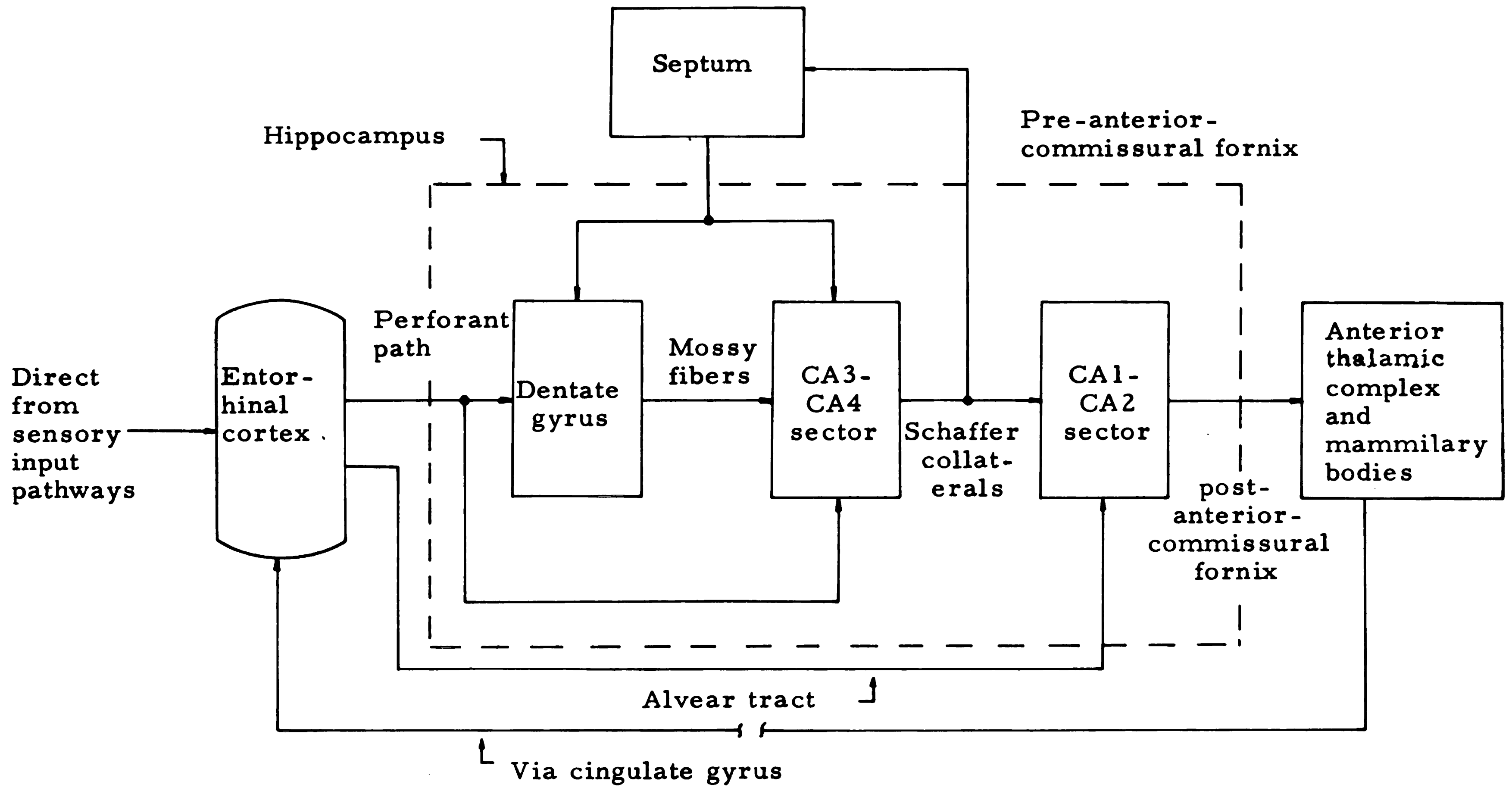


Figure 1. 2. 1. Dorsal hippocampus and connections.

A schematic of the dorsal hippocampus and its connections to other structures in the rat. Note that the hippocampus has been partitioned into three areas, based on morphology. They are the dentate gyrus, CA3-CA4, and CA1-CA2. All three areas appear to contain recurrent inhibition, but CA3 has the best known structure. The path from CA3 to septum and back to CA3 will play an important part in subsequent discussion. For details see Kilmer [12], from which this figure has been adapted.

Firin
Rate

Pyrami
Cell P5

Basket
Cell B4

Pyram
Cell P3

Basket
Cell B1

Figu

All pyr

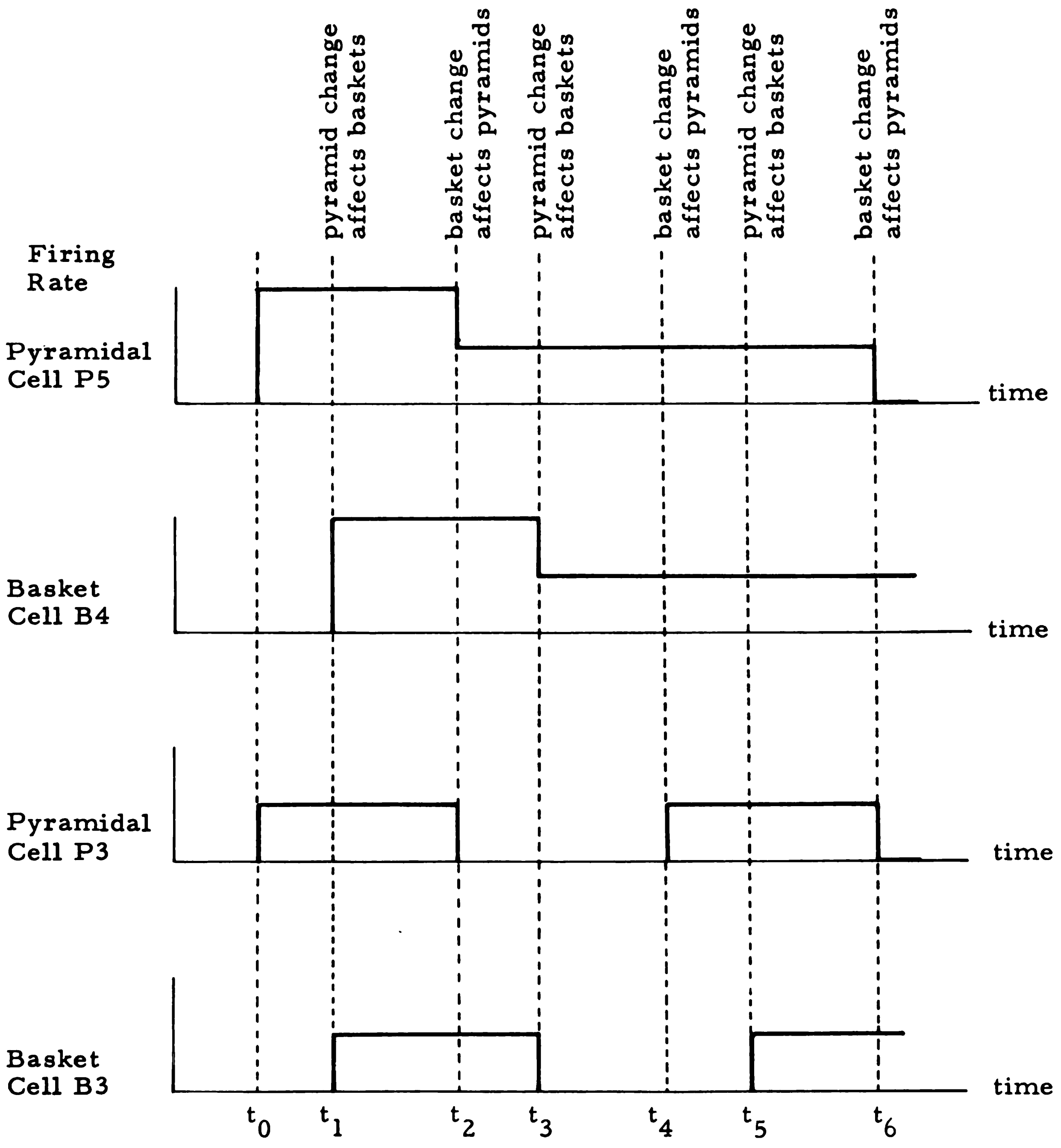

pyramid change affects baskets
basket change affects pyramids
pyramid change affects baskets
basket change affects pyramids
pyramid change affects baskets
basket change affects pyramids
Firing Rate
Pyramidal Cell P5
time
Basket Cell B4
time
Pyramidal Cell P3
time
Basket Cell B3
time
t_0
t_1
t_2
t_3
t_4
t_5
t_6

Figure 1. 3. 1. A phase diagram for the example given in Section 1. 3.

All pyramid and basket numbers refer to Figure 1. 1.

remair
basket
B4 re

paper
for sy
is ur
exce
justi
recu

som
whic
the

1.4

car
Pr
the
be
m
wl
cc
is
u
ti

i
I
1

remains unchanged. At time t_5 these changes will be felt by the basket cells; as a result suppose B3 returns to a moderate rate and B4 remains unchanged. And so on ad nauseam.

In order to circumvent these nasty expositional problems, this paper has developed a formal language, called functal system theory, for systems of the kind exemplified by the hippocampus. The reader is urged not to become discouraged by what may seem to him to be excessive formalism in the following chapters; the formalism is justified by the compactness with which it expresses functions involving recurrent inhibition.

In addition to modeling the hippocampus as a functal system, some of the operational principles of the class of neuromime nets to which the model belongs are also given, along with characteristics of the output sequences of such nets.

1.4. What is the Function of the Hippocampus?

An hypothesis of the primary task of the mammalian hippocampus has been proposed by W. Kilmer and T. McLardy [12]. Previously, Kilmer and W. McCulloch [13] proposed that the task of the mammalian reticular formation is to decide the basic mode of behavior of an animal. A mode might be to fight, take flight, groom, mate, or eat. It is plausible to suggest that another structure exists which takes modal and current sensory information and generates commands for acts within modes. For instance, if the mode decision is to fight, another structure may select the tactics or style to be used. Kilmer and McLardy believe that the hippocampus is part of this structure, at least during the animal's behavior-formative period.

Functal system theory is used in this paper to provide an interpretation of the hippocampus's function which supports this hypothesis. In a few words, the interpreted function is trainable recurrent inhibition.

CHAPTER 2

FUNCTAL SYSTEMS

2. 1. An Informal Description of the Functal System

The hippocampus and its associated structures appear to be related to a theoretical structure called a functal system. Informal definitions of each of the components and of the overall operation of the system are as follows (see Figure 2. 1. 1).

1. The INPUT GENERATOR interprets the present environment according to its built-in predisposition and produces an input from a finite set of possible inputs.

2. The INPUT BUFFER, which is under the control of the TRAINER, performs a combinational circuit transformation on the input and produces the input to the functal net.

3. The TRAINABLE FUNCTAL NET has these characteristics:

   a. It is an array of three types of elements: functals, function generators, and delays.

   b. Each functal is capable of realizing any one of many functions. Each function being realized is under the control of the trainer.

   c. A fixed connectivity exists between the elements of the net (the rules defining the connectivity may involve probability density functions).

The functal net generates a finite sequence of outputs for a given single input.

4. The TARGET TABLE GENERATOR has observed the environment by this time and has constructed a target table.

5. The TARGET TABLE contains the functions that each of the functals is required to generate. All communication with the target table is controlled by the READ/WRITE HEAD AND CONTROLLER.

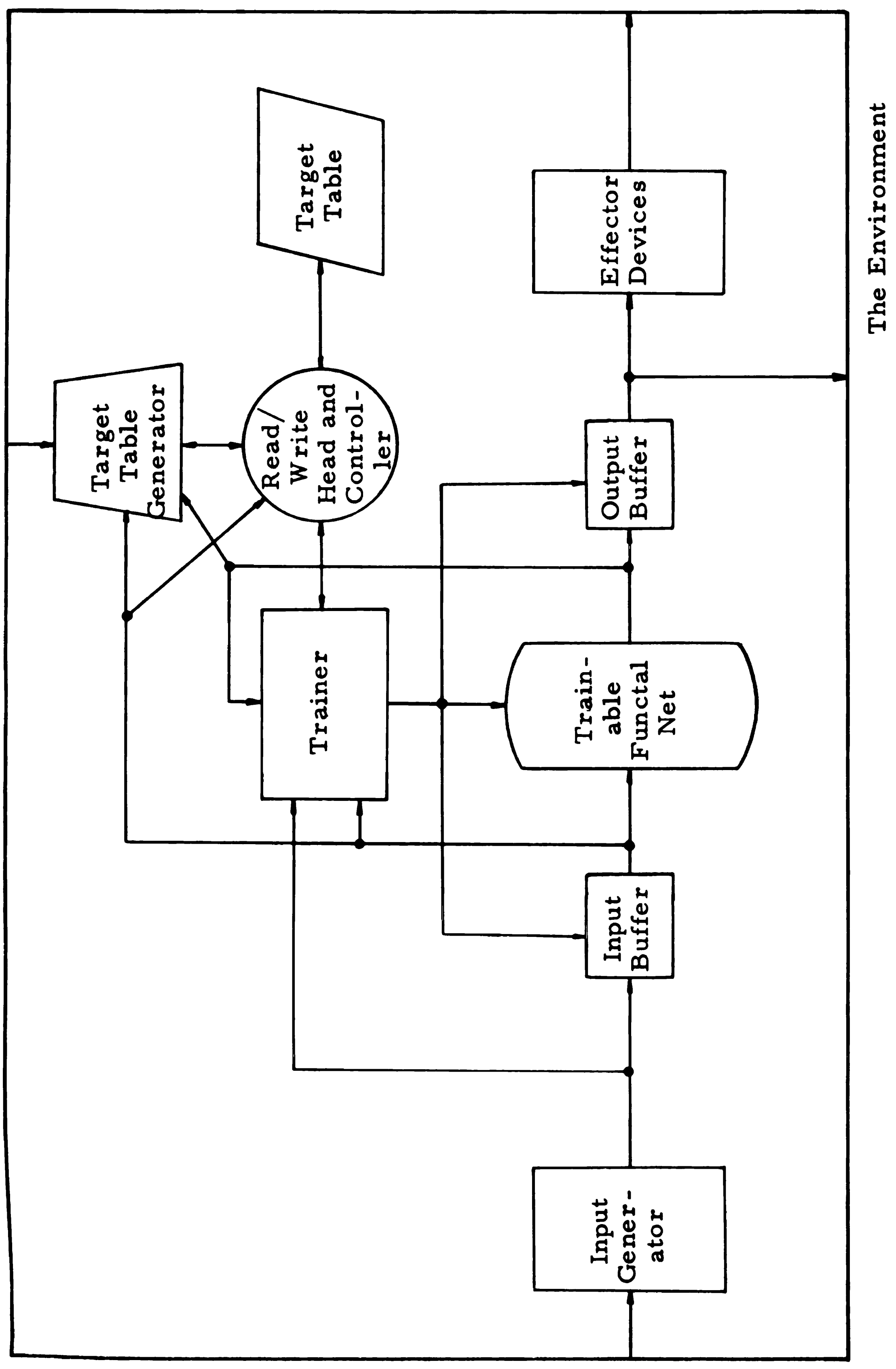


Figure 2. 1. 1. The functal system surrounded by an environment.

compu
requir

the co

buffer
It may

the re
assum
algori

2. 2.

(that
of a f
doma
if tim
Defin

and a

where

Defin

elem
exter
Defin

is the

6. The TRAINER compares the desired output with the output computed by the net and corrects any functals not generating the required output.

7. The OUTPUT BUFFER is combinational circuitry under the control of the trainer.

8. The EFFECTOR DEVICES use the output from the output buffer to allow the entire system to interact with the environment. It may also be true that this output affects the environment directly.

A formal discussion of functal system theory is presented in the remainder of this chapter. Throughout the discussion it will be assumed that the functal net and all its associated structures and algorithms operate synchronously in discrete time.

2. 2. The Functal

The intuitive concept of a functal is that it is a mechanism (that is, an algorithm or physical device) which can realize any one of a finite number (greater than 1) of different functions. If the domains and ranges of the functions are assumed to be finite sets, and if time is assumed discrete, then:

Definition 2. 2. 1

A functal can be represented over all time by

$$\langle \mu, \gamma, \mathcal{F}, \Sigma \rangle$$

and at any time t by

$$\sigma(t) = F_i(M(t), Z(t), t)$$

where

$$\sigma(t) \in \Sigma, \quad F_i(\cdot) \in \mathcal{F}, \quad Z(t) \in \gamma, \quad \text{and} \quad M(t) \in \mu .$$

Necessary supporting definitions are:

Definition 2. 2. 2

μ, a finite set, is the controlled input set of the functal. The elements of μ are called controlled inputs. (μ is under direct external control.)

Definition 2. 2. 3

$$\gamma = \{ Z(t) = Z^1(t)\, Z^2(t) \ldots : Z^i(t) \text{ is an element of internal input set} \}$$

is the internal input sequence set. The elements of γ are called

internal
that car
Definiti

is the f
can rea
Definit

is the
Defini

Defini

of an
an ou
Defir

elem
Defir

is ca

empl
func
gene
of in
even

internal inputs. (γ takes into account possible inputs to the functal that cannot be directly controlled.)

Definition 2.2.4

$$\mathcal{F} = \{ F_i : \mu \times \gamma \xrightarrow{F_i} \Sigma \}$$

is the function set. (The function set contains the functions the functal can realize.)

Definition 2.2.5

$$\Sigma = \{ \sigma(t) = H^1(t)\, H^2(t) \ldots : H^i(t) \text{ is an element of the finite output set } \mathcal{H} \}$$

is the set of output sequences.

Definition 2.2.6

Any member of Σ, $\sigma(t)$, is called an output sequence.

Definition 2.2.7

Any vector element

$$H^i(t) = \begin{bmatrix} h^i_1(t) \\ h^i_2(t) \\ \vdots \\ h^i_n(t) \end{bmatrix}$$

of an output sequence is called an output sequence element (or simply an output).

Definition 2.2.8

A component $h^i_j(t)$ of an output vector is called an output element.

Definition 2.2.9

The sequence

$$\sigma_j(t) = h^1_j(t)\, h^2_j(t) \ldots$$

is called an output sequence component.

The specification of both controlled and internal inputs emphasizes a basic property of functals: only controlled inputs to a functal are provided by the input generator; internal inputs are generated within the functal net. In particular, feedback is one kind of internal input. If it is present the functal can generate a sequence, even when there is only a single input from the input generator.

2.3. T

Definit

the out
at the c
these.

outputs
configu
unit de
delays

the bel

Defini

called
be Q
with t
with t
Z is

even l
by co
compo
of Z

Defini

the co

of the

2.4.

under
Assur

2.3. The Functal Net

Definition 2.3.1

A functal net consists of functals plus unit delay elements at the output of each functal, function generators plus unit delay elements at the output of each generator, and a connectivity scheme relating these.

The delay elements are included for two reasons. First, the outputs of some of the elements in a functal net will be in a feedback configuration. The standard way to analyze such nets is to insert unit delays. Second, all physically realizable functal nets will have delays in lines and elements.

The concept of state plays a fundamental role in understanding the behavior of functals:

Definition 2.3.2

The outputs of the delay elements can be ordered in a vector called the state vector $Q \in \mathcal{Q}$, the state vector set. The ordering will be $Q = (X, Z)$, where X is the output of the delay elements associated with the functals and Z is the output of the delay elements associated with the function generators. X is called the functal state vector and Z is called the generator state vector.

If the functal net is considered to be a single functal of an even larger net, then the elements H of the output sequence set will, by convention, have the form $H = (H_f, H_g)$, where H_f are those components of X considered outputs and H_g are those components of Z considered outputs.

A special kind of functal net is the trainable functal net:

Definition 2.3.3

A trainable functal net is a functal net whose function is under the control of a defined structure called the trainer.

The trainer will be discussed in more detail after the character of the input-output relationship of a general functal net is revealed.

2.4. The Concepts of State and Output Foundations

There is a graphic viewpoint which can promote some initial understanding into the design and analysis problems for functal nets. Assume that at some initial time t_o the vector of functions currently

realized
the stat
<u>Definiti</u>

is calle
<u>Definit</u>

combi
For ea
new st
these
<u>Defini</u>

trary
<u>Defin</u>

funct

vecto
the c
<u>Defi</u>

a co
<u>Defi</u>

a gi

2. 5
2. 5

nets
<u>Def</u>

it i

realized by the functals is $F(t_o)$, the controlled input vector is $I(t_o)$, the state vector is $Q(t_o)$, and the output vector is $H(t_o)$.

Definition 2.4.1

The quadruple

$$L(t) = \langle F(t), Q(t), I(t), H(t) \rangle$$

is called the locus of the functal net at time t.

Definition 2.4.2

The locus $L(t_o)$ is called the initial locus.

Consider each and every combination of F and I. Within each combination, place the net in every possible state in the state set. For each state allow the net to compute for one period and record the new state. Construct a standard state table or state diagram from these data. The resulting representation is given a special name.

Definition 2.4.3

A state level is the state structure associated with any arbitrary but specified combination of F and I. The notation is $l(F, I)$.

Definition 2.4.4

The set of all state levels is called the state foundation of the functal net.

The fact that the output vector H is a subvector of the state vector Q can be used to construct an equivalent set of definitions for the output.

Definition 2.4.5

An output level $lo(F, I)$ is the output structure associated with a combination of any F and any I.

Definition 2.4.6

An output foundation is the set of all possible output levels for a given functal net.

2.5. Properties of Output and State Levels

2.5.1. Properties of State Levels

Kauffman [14] has demonstrated the following property for nets of arbitrarily connected switching elements having fixed inputs:

Definition 2.5.1

A state with the property that the net remains in the state once it is entered is called an equilibrium state.

Definiti

called a

Definiti

leads t

Proper

proper

of arb

follov

Defin

same

state

impo

Defi

star

Pro

Pro

Def

wit

cal

Definition 2. 5. 2

A subsequence of states that is continuously repeated is called a state cycle.

Definition 2. 5. 3

A subsequence of states with the property that it eventually leads to a state cycle is called a state run-in.

Property 2. 5. 1

Each state of a level has one and only one of the following properties:

1. It is in a state run-in.
2. It is an equilibrium state.
3. It is in a state cycle.

It is clear that this property is true for function generators of arbitrary but finite domains and ranges.

A useful relation between state run-ins and state cycles is the following.

Definition 2. 5. 4

Within a state level, all states belonging to run-ins to the same cycle plus all the states belonging to the cycle form a set of states called the state cycle complex.

A typical state level is shown in Figure 2. 5. 1.

One state in each state cycle complex will assume particular importance:

Definition 2. 5. 5

Any state in a state cycle complex may be designated as a start state for the cycle.

Property 2. 5. 2

There can be only one start state per state cycle complex.

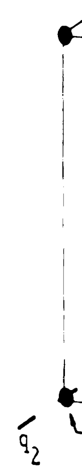
Property 2. 5. 3

A given start state will lead to a unique state cycle.

Definition 2. 5. 6

A sequence of states (run-in plus cycle) in state level $l(F, I)$ with start state q_o will be denoted by $C(q_o, F, I)$, and will be called a state sequence.

An important property of any state sequence is:



The s
value

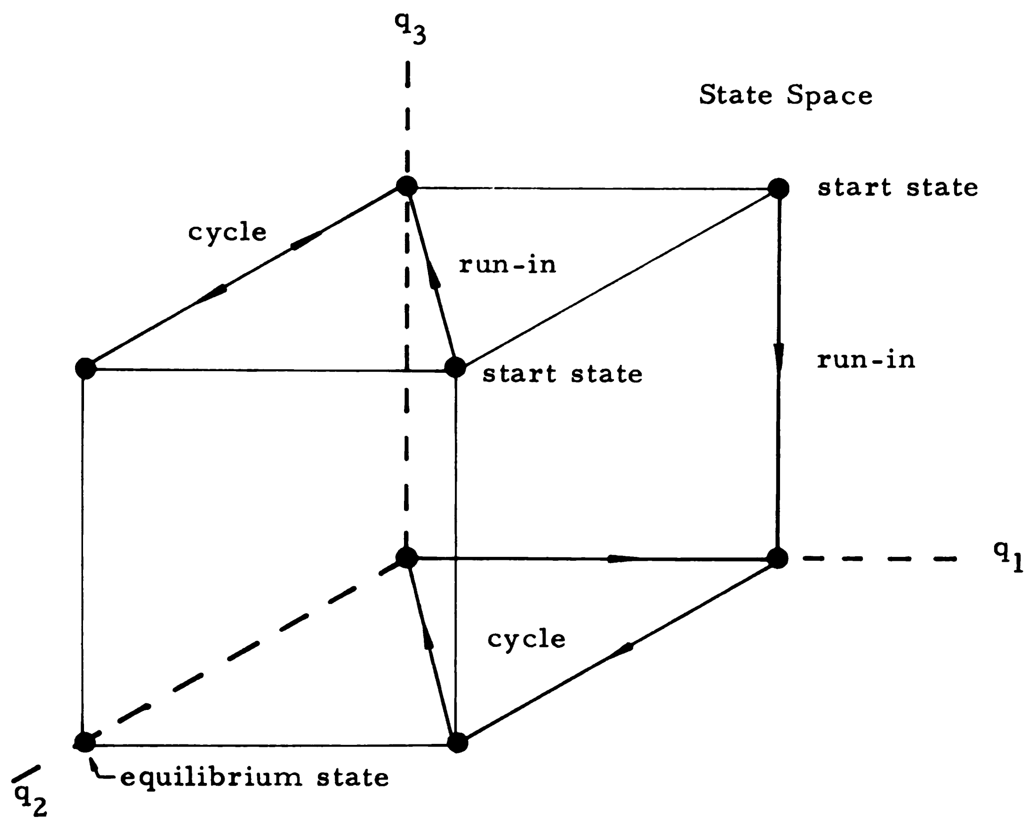


Figure 2. 5. 1. A typical state level structure.

The state space is three-dimensional, with each state being binary valued.

Property

T

indicate

second p

2.5. 2.

for eve

Definiti

called a

state r

Definit

is call

Defini

equili

is:

Defin

same

outpu

Defin

outp

$\sigma(H^1$

funct

two

Property 2.5.4

The second occurrence of any state in the sequence $C(q, F, I)$ indicates the completion of the state cycle and the beginning of a second pass through the cycle.

2.5.2. Properties of Output Levels

Since the output vector H is subvector of the state vector Q, for every state cycle there is a corresponding output cycle:

Definition 2.5.7

A sequence of outputs which continuously repeats itself is called an output cycle.

There will also be sequences of outputs corresponding to the state run-ins:

Definition 2.5.8

A sequence of outputs which eventually leads to an output cycle is called an output run-in.

Corresponding to the equilibrium state:

Definition 2.5.9

An output which continuously repeats itself is called an equilibrium output.

Finally, the definition corresponding to the state cycle complex is:

Definition 2.5.10

Within an output level, all outputs belonging to run-ins to the same cycle plus all the outputs belonging to the cycle form a set of outputs called the output cycle complex.

Definition 2.5.11

An output cycle plus output run-in in level $lo(F, I)$ with initial output H^1 is an output sequence of the net and is signified by $\sigma(H^1, F, I)$.

Of course the similarity between the output sequence of the functal definition and the above definition is no accident. Indeed,

$$\sigma(H^1, F, I) = H^1H^2H^3 \ldots = F(I, Z).$$

Now, consider a specific state level $l(F, I)$ and the following two state cycles:

(The cycles are listed in the form

$$\begin{array}{llll} q_1(t_1) & q_1(t_1+d) & q_1(t_1+2d) & \cdots \\ q_2(t_1) & q_2(t_1+d) & q_2(t_1+2d) & \cdots \\ \vdots & & & \\ q_n(t_1) & q_n(t_1+d) & q_n(t_1+2d) & \cdots \end{array}$$
)

cycle number 1	cycle number 2
0 0 0 0 0	0 0 0 0
1 0 0 1 1	1 0 1 1
1 1 0 0 1	1 1 0 1
0 0 0 0 0	1 1 1 1

If q_1 and q_2 are defined as the outputs, then the sequences $\begin{smallmatrix}0&0&0&0&0\\1&0&0&1&1\end{smallmatrix}$ and $\begin{smallmatrix}0&0&0&0\\1&0&1&1\end{smallmatrix}$ are the output cycles. Note that the $\begin{smallmatrix}0\\0\end{smallmatrix}$ and $\begin{smallmatrix}0\\1\end{smallmatrix}$ outputs are in more than one output cycle and the second occurrence of $\begin{smallmatrix}0\\0\end{smallmatrix}$ in cycle number 1 did not signal the end of the cycle. These observations can be generalized in the following properties:

Property 2. 5. 5

Any single output may be in more than one output cycle complex.

Property 2. 5. 6

It is not possible to determine the end of an output cycle by comparing the current output with previous outputs.

These two properties play a significant role in determining the complexity of the functal system trainer.

2. 6. The Target Table Generator

A target table can be generated whenever it is both advantageous to do so and conditions permit. This generally requires the target table generator to know what functions can be trained from each function the net can realize. In other words, the target table generator should have available as a reference the following class of sets:

Definition 2. 6. 1

$$\mathcal{C} = \{\mathcal{S}_i : \mathcal{S}_i \text{ is a convergence set}\}$$

is the convergence class.

Defini

is the

table
requir
that th
just c
each t

On the
of the
in par

2. 7.

sequen
Defini

called

sequen
target
Definit

sequen
Definit

corres
Definit

sequenc

Definition 2. 6. 2

$$\mathcal{B}_i = \{ F_k \in \mathcal{F} : \text{the functal realizing the function } F_i \in \mathcal{F} \text{ can be trained to realize the function } F_k \}$$

is the convergence set of the function F_i.

Implicit in these definitions is the requirement that the target table generator must also have knowledge of the set $\mathcal{F}$. This requirement should not be taken lightly. In the real system it implies that the target table generator and the functal net must be more than just casually related: they must have evolved in a way that allows each to know what it can expect from the other.

> This kind of relationship could come about very naturally in a neural system if the target table generator structure grew the functal net to perform a deligated task.

On the other hand, in the design of the artificial system, the design of the target table generator and the functal net will have to proceed in parallel.

2. 7. The Target Table

The target table contains a list of output sequences, one sequence for every possible input to the functal net.

Definition 2. 7. 1

When contained in a target table, an output sequence will be called a target sequence, with the notation $\sigma^*(t)$.

As with the output sequence, the target sequence is a vector sequence. The following definitions locate the various parts of the target sequence.

Definition 2. 7. 2

A target sequence element $H^{j*}(t)$ corresponds to the output sequence element of Definition 2. 2. 7.

Definition 2. 7. 3

A target sequence component element, or atom, $h_i^{j*}(t)$, corresponds to the output element of Definition 2. 2. 8.

Definition 2. 7. 4

A target sequence component $\sigma_i^*(t)$ corresponds to the output sequence component of Definition 2. 2. 9.

Defi

term

fınct

leng

com

mod

a ta

Supp

targ

2. 8.

2. 8.

algo

desc

eithe

Definition 2.7.5

The set of target sequence components for a single output terminal and over all possible input values to the net is called a functal section of the target table.

In order to keep the target table as compact as possible, the length of a target sequence is limited to the maximum length of any component's run-in plus cycle. Along with this convention, a modified regular expression notation is used when explicitly listing a target sequence. This notation is best defined by example. Suppose the functal net has four outputs and the components of the target sequence for some input I are:

$$\sigma_1^*(I) = 0123456456456.....$$

$$\sigma_2^*(I) = 0000000.....$$

$$\sigma_3^*(I) = 234523452345.....$$

$$\sigma_4^*(I) = 8722222222222.....$$

Then the notation for these is:

$$\sigma_1^*(I) = 0123456(456)^*$$

$$\sigma_2^*(I) = 00^*$$

$$\sigma_3^*(I) = 2345(2345)^*$$

$$\sigma_4^*(I) = 8722^*$$

As one target sequence, the notation is:

$$\sigma^*(I) = \begin{matrix} 0123 \\ 0000 \\ 2345 \\ 8722 \end{matrix} \begin{bmatrix} 456456456456 \\ 000000000000 \\ 234523452345 \\ 222222222222 \end{bmatrix} \begin{bmatrix} 456456456456 \\ 000000000000 \\ 234523452345 \\ 222222222222 \end{bmatrix}^*$$

2.8. A General Training Structure and Algorithm

2.8.1. Movement Within Foundations Under Input and Function Change

In this section a general form for a trainer structure and algorithm is proposed. First, though, it will be necessary to describe what happens in the foundations when there are changes either in the inputs to a net or in the functions of a net.

Assume that the initial locus of the net is

$$L(t_o) = \langle F(t_o), I(t_o), Q(t_o), H(t_o) \rangle .$$

Therefore, the net is in:

a. state level $l(F(t_o), I(t_o))$,
b. state sequence $C(Q(t_o), F(t_o), I(t_o))$,
c. output level $lo(F(t_o), I(t_o))$,
d. output sequence $\sigma (H(t_o), F(t_o), I(t_o))$.

Assuming $I(t_o)$ and $F(t_o)$ are not changed, (b) and (d) define the future of the net.

Now suppose the input is changed and is effective at time t_1. At this time the net is in state $Q(t_1)$ and this becomes a new start state. This means that the net is in:

a. state level $l(F(t_o), I(t_1))$,
b. state sequence $C(Q(t_1), F(t_o), I(t_1))$,
c. output level $lo(F(t_o), I(t_1))$,
d. output sequence $\sigma (H(t_1), F(t_o), I(t_1))$.

Finally, suppose the function that the net is realizing is changed and becomes effective at time t_2. At this time the net is in state $Q(t_2)$ and this becomes a new start state. Therefore, the net is in:

a. state level $l(F(t_2), I(t_1))$,
b. state sequence $C(Q(t_2), F(t_2), I(t_1))$,
c. output level $lo(F(t_2), I(t_1))$,
d. output sequence $\sigma_i(H(t_2), F(t_2), I(t_1))$.

2.8.2. The Operation of the Trainer Under Input or Function Change

Suppose the target table has entries $\sigma^*(I(t_o))$ and $\sigma^*(I(t_1))$. The ideal situation would be that when the input changes,

$$\sigma^*(I(t_1)) = \sigma (H(t_1), F(t_o), I(t_1)).$$

Of course, in general there is no assurance that this will happen. The structure in Figure 2.8.1 is proposed to insure that the ideal situation will occur, at the cost of some "dead time" when the input changes. The function of the components of the structure are:
INPUT-CHANGE DETECTOR: Detects changes in the input.

Inp
Ch
Det
F
In
Gen

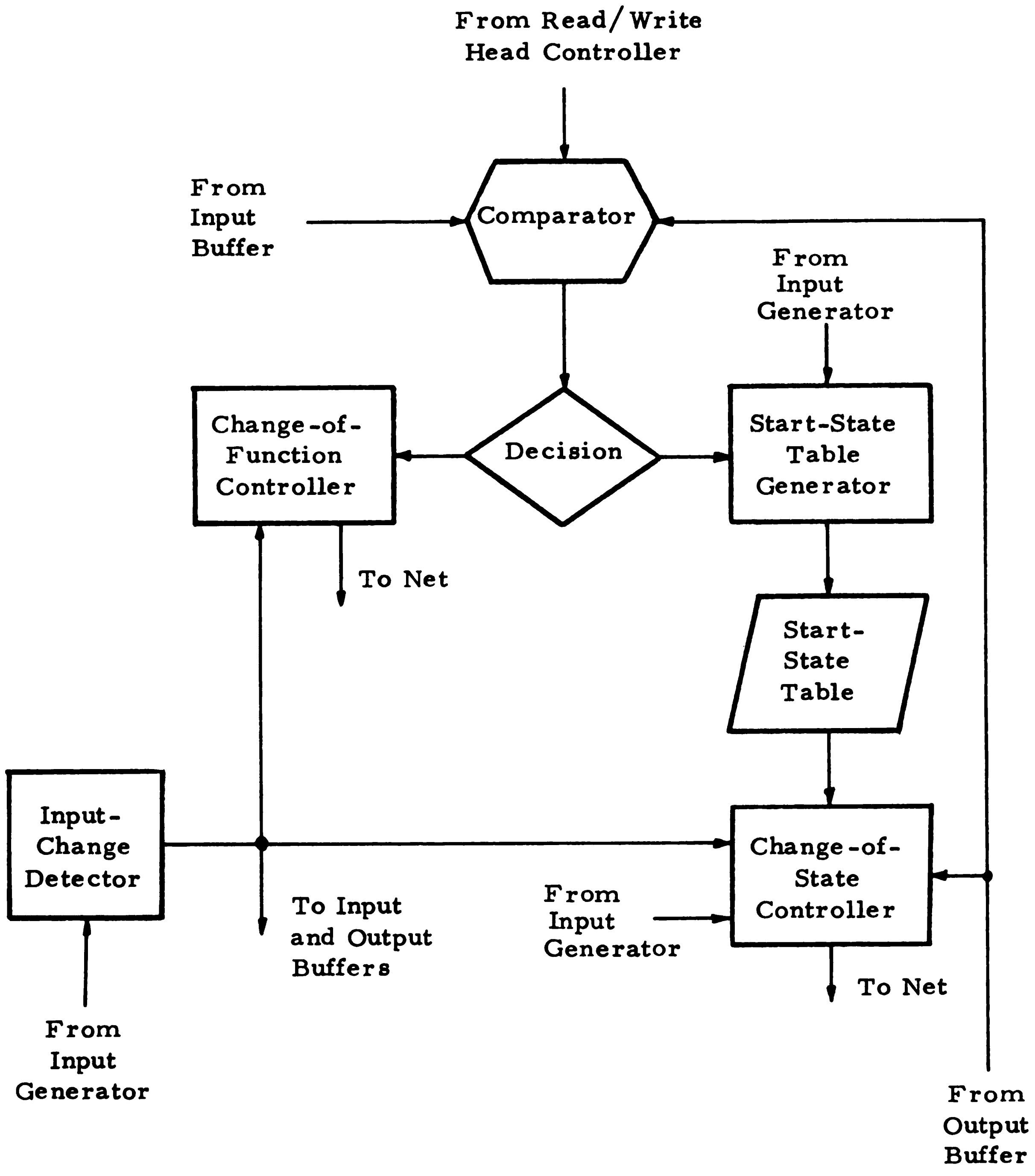


Figure 2.8.1 The basic trainer structure of a functal system.

STAR

CHA

COM

DEC

CHA

STA

Algo

beco

sign
outp
nece
outp
para
stat

by th
diffe
deci
chan

net t
reali

START-STATE TABLE: Contains a list of start states, one for each possible input.

CHANGE-OF-STATE CONTROLLER: Retrieves a start state from the start-state table and places the net in that state.

COMPARATOR: Uses the inputs it receives to compare the output it receives from the net with the appropriate entry in the target table. It may also be necessary to signal the comparator to suspend operations until the output associated with a new input has made its way to the comparator.

DECISION CONTROLLER: When an error has been detected, this component decides whether to change the start state or the function.

CHANGE-OF-FUNCTION CONTROLLER: Changes the function being realized by the net.

START-STATE TABLE GENERATOR: Changes the start-state table.

The algorithm the structure follows is:

Algorithm 2.8.1

1. If the input changes, the input change detector output becomes one.

2. This activates the change-of-state controller, which signals the input buffer to withhold the new input from the net and the output buffer to generate some "neutral" output. (It may also be necessary to signal the comparator to suspend operations until the output associated with the new input has found its way to the comparator.) Then the change-of-state controller consults the start-state table and imposes a new start state on the net.

3. The comparator is continually comparing outputs generated by the net with the desired outputs in the target table. If there is a difference between any two of them, the comparator notifies the decision controller. This controller, in turn, activates either the change-of-function controller or the start-state table generator.

4. The change-of-function controller attempts to train the net to the sequence in the target table by changing the function being realized by the net.

5. The start-state table generator attempts to train the net

to the sequence in the target table by changing the start state of the input in question.

The detailed manner in which the decision controller, the change-of-function controller, and the start-state table generator operate is, of course, of utmost interest. However, very little more can be said about these devices without having a particular situation in mind, such as the hippocampus and its environment.

Two kinds of time intervals will be of interest to the trainer.

Definition 2. 8. 1

The time period is the time during which one output is computed by the net.

As mentioned previously, when there is an output error, the generation of outputs is suspended and training takes place. During training the passage of time is indicated by the number of time pulses that occur.

Definition 2. 8. 2

The time pulse is the fundamental time quantum of the net.

2. 9. The Trainer, Target Table Relationship

There are two methods the trainer might use to compare an output sequence with a target sequence. Either it can wait until the entire sequence has been generated and then compare it with the target sequence, producing a single judgment on the similarity of the two sequences; or it can compare the outputs one at a time, producing judgments after each output. The first alternative is undesirable because it implies that a sequence is equivalent to a single net output. Consequently:

Assumption 2. 9. 1

Each output of the functal net is assumed to have an effect on the environment and/or the effector devices. Furthermore the trainer has the capability to compare any single output with the corresponding entry in the target table.

Since the target sequence consists of a run-in plus a cycle, it might be efficient to compare the output sequence with the target sequence only until the completion of the first output cycle.

After that, and for as long as the input does not change, the output and corresponding target sequence element would no longer be compared. There are two reasons, however, that this approach is undesirable. First, from Property 2. 5. 6 the generated sequence may be a subsequence of a longer sequence. Second, no restrictions are placed on when the target table can be changed. Therefore,

Assumption 2. 9. 2

The trainer has a mechanism for detecting the end of an output cycle and a mechanism for finding the beginning of the cycle in the target sequence representation whenever the end is detected.

Example 2. 9. 1

Each target sequence $\sigma^*(t)$ can be visualized as being on an erasable tape. A mark on a companion tape can be used to indicate the end of the run-in and the beginning of the cycle. The read/write head and controller can be assumed to consist of two parts: a read-head and its controller, and the write-head and its controller. The read-head controller could initialize the read-head at the first element of the tape whenever a new input is detected and it could move the read-head to the right one element every time period. If the final element in the representation is read and the input has not changed, then the controller could move the read-head back (to the left) in the tape until it found the beginning-of-cycle indicator on the companion tape.

2. 10. Measures of Functal Net Performance

It is important to have measures by which nets, natural or man-made, can be compared. Some of the obvious measures are the number of functals, the number of function generators, the similarity of connectivity, and the similarity of the training algorithm. Others are:

1. The convergence set. One measure of the versatility of a functal net is the number of net functions that the net can be trained to realize from a specified function. This measure is the cardinality of the convergence set of the specified function.

2. The statistics of output sequence lengths. Another class of measures of interest to some studies is the statistics of the output sequence lengths. For instance, it might be of importance to look at the distribution of run-in, cycle, or equilibrium lengths or the percentage of sequences in the three categories.

3. The statistics of convergence times. It will not be of much value to be able to train a functal net if a prohibitively long time is required for training. Hence, the following measure.

Definition 2. 10. 1

The convergence time $T = T(\sigma(I), \sigma^*(I))$ is the number of time pulses required to train a functal net to produce $\sigma^*(I)$ when it was originally producing $\sigma(I)$.

Some of the convergence time statistics are:

a. The longest training period for a convergence set.
b. The shortest training period for a convergence set.
c. The longest training period for the entire convergence class.
d. The shortest training period for the entire convergence class.
e. The average convergence times for any of the above.

2. 11. The Functal System Analysis Problem

The following procedure has been found useful in modeling and analyzing a natural system (the hippocampus) by a functal system.

Step 1. In the natural system identify:

a. The input and output spaces.
b. The functals.
c. The function generators.
d. The rules for connectivity.
e. The delay times between and within elements.
f. The trainer and its algorithm.
g. The target table generator and its algorithm.

Step 2. Specify the model--equations and parameters--of each of the above.

Step 3. If possible, isolate the physical elements and establish that the element models are satisfactory; that is, the input-output transformations are within an acceptable range of those found in the physical system. Go back to Step 2 or perhaps even Step 1 if changes need to be made in either the model or the natural system concepts.

Step 4. Verify that the model behaves in the way the modeler (you) expects it to. In some cases this may lead to a jump to either Step 2 or Step 7.

Step 5. Record properties and measure the characteristics of the model.

Step 6. Design experiments or interpret existing data to obtain comparable properties and characteristics of the physical system.

Step 7. Compare the results of Steps 5 and 6. If the comparison is poor, then make a judgment as to the cause and return to the appropriate previous step to revise either the physical system concepts or the model.

It should be emphasized that Step 7 may lead to entirely new concepts of the physical system. This is one of the two main contributions functal net modeling (or any modeling, for that matter) can make. The other contribution is that the functal net concepts are so designed that computer component analogs can be constructed of the physical system; this, especially when the physical system is neural in nature, would lead to entirely new kinds of machines.

CHAPTER 3

A FUNCTAL SYSTEM MODEL OF THE HIPPOCAMPUS. PART 1

3. 1. Introduction

The hippocampus system model presented in this chapter and Chapter 5 (see Figure 3. 1. 1) is the compromise design which resulted from intense negotiations to simultaneously satisfy four competing interests: The anatomical and neurophysiological data on neurons and nervous systems in general and the hippocampus in particular, the ability to meaningfully simulate the model either in hardware or software, the experimenter's intuition, and the functal system theory.

The discussion of the model has been divided into two parts. The first part, this chapter, discusses the model of the hippocampus CA3 sector and those other elements of the canonical functal system which do not rely on certain theoretical properties of the CA3 model. These include the input and output sets and buffers. The next chapter develops some of the computation theory of the CA3 model and then Chapter 5 completes the specification of the hippocampus system model.

3. 2. The Input and Output Sets of the Hippocampus System Model

There is insufficient evidence from the firing rate data of the mossy fiber input or the pyramidal cell output of the CA1 sector to determine the significant ranges for the input and output variables of any hippocampus system model. It seems reasonable, therefore, to choose the simplest range possible, the binary set (low firing rate, high firing rate). The corresponding model values will be (0, 1).

3. 3. The Input Buffer

The input buffer defines the connectivity of each component I_j of the system input vector I to each pyramidal cell analog k in the

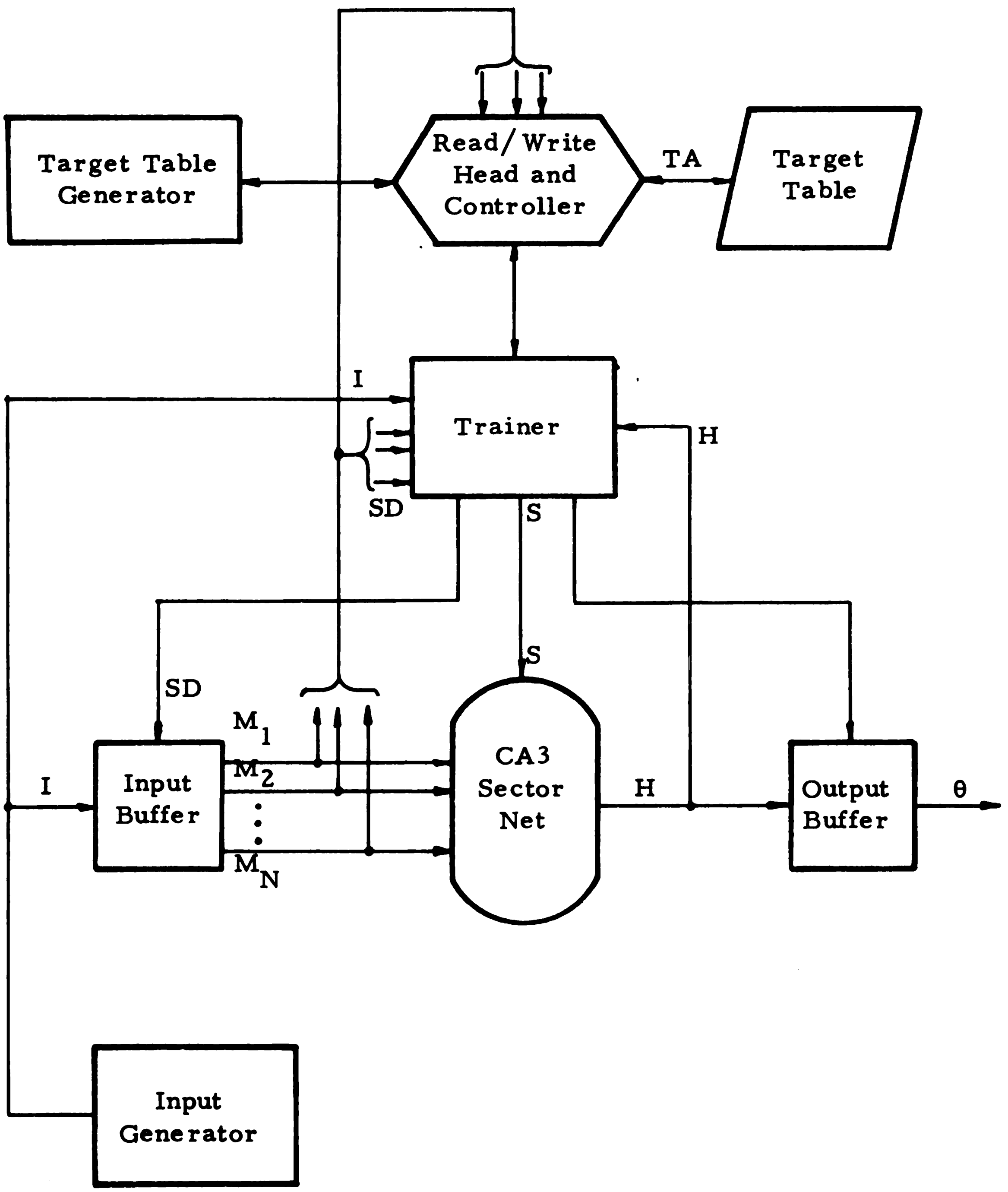


Figure 3.1.1. The overall structure of the hippocampus system model.

CA3 model. It also acts as a combinational switching circuit with Table 3. 3. 1 as the truth table (where M is the input matrix to the CA3 sector model). From the table it is clear that SD is used to do one of three things: block the input (SD = 0), allow the input to pass unchanged (SD = 1), or amplify the input (SD = 2).

In the hippocampus itself, it is possible that the dentate performs the input buffer function, with SD corresponding to the septal input, I corresponding to the perforant path fibers, and M corresponding to the mossy fiber output of the dentate.

3. 4. The CA3 Sector Net

3. 4. 1. Introduction

The CA3 sector net is intended to be a model of the CA3 sector of the hippocampus. The major elements in the net are the pyramidal cell logic units (PCLUs), which are functals representing the pyramidal cells; and the basket cell logic units (BCLUs), which are function generators representing the basket cells. The inputs to the net are the matrix M and the vector S. The latter is assumed to correspond to the septal input. The output of the net, the vector H, represents the Schaffer and fimbrial fornix collaterals. The values of the elements of the output are taken from the set {0, 1, 2}, where 0 is assumed to represent a low firing rate, 1 a medium firing rate, and 2 a high firing rate. The design rationale for the net is given in Appendix A.

3. 4. 2. The Pyramidal Cell Model: The Pyramidal Cell Logic Unit

This functal, which is a close kin to the adaptive threshold logic unit so prevalent in the literature of neural modeling, is designed to generate a digital approximation to the firing rate of a pyramidal cell.

> The primary difference between the adaptive threshold logic unit and the PCLU is that an individual weight of a PCLU can only be adjusted in one direction.

There are two major parts to the PCLU, the discriminant function and the training algorithm.

Table 3. 3. 1.

The Truth Table for the Input Buffer of the Functal Systems Model of the Hippocampus

SD	I_j	Does a connection exist between input I_j and k?	$m_{kj} \epsilon M$
0	0	.	0
0	1	.	0
1	0	.	0
1	1	yes	1
1	1	no	0
2	0	.	0
2	1	yes	2
2	1	no	0

Definition 3. 4. 1

The discriminant function of PCLU i, denoted by

$$\lceil A_i(t) \cdot M_i(t) - B_i(t) \cdot Z_i(t) \rceil_{T_{12}} = y_i(t), \quad \text{is}$$

$$T_2 > A_i(t) \cdot M_i(t) - B_i(t) \cdot Z_i(t) \geq T_1 \quad \text{iff} \quad y_i(t) = 1$$

$$A_i(t) \cdot M_i(t) - B_i(t) \cdot Z_i(t) \geq T_2 \quad \text{iff} \quad y_i(t) = 2$$

$$A_i(t) \cdot M_i(t) - B_i(t) \cdot Z_i(t) < T_1 \quad \text{iff} \quad y_i(t) = 0$$

where

$m_{ij}(t) = 1$ or 2 implies $a_{ij}(t)\, m_{ij}(t) = a_{ij}(t)$.

The vectors and constants in the discriminant function have been given names:

Definition 3. 4. 2

The vector $A_i \epsilon\, \mathcal{P}_m$ is the vector of mossy fiber (mf) weights for PCLU i.

Definition 3. 4. 3

The vector $B_i \epsilon\, \mathcal{P}_n$ is the vector of feedback weights for PCLU i.

Definition 3. 4. 4

The vector $W_i = (A_i, B_i)$ is the weight vector for PCLU i.

Definition 3. 4. 5

The vector $M_i \epsilon\, \mathcal{T}_m$ is the set of mossy fiber (mf) inputs to PCLU i. (M_i is a row of the matrix M.)

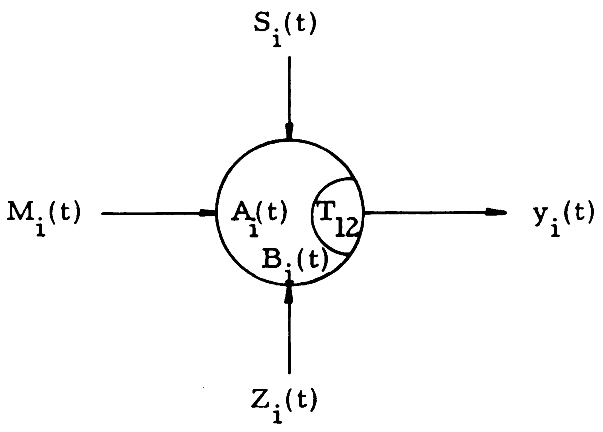
Definition 3. 4. 6

The vector $Z_i \epsilon\, \mathcal{B}_n$ is the set of feedback inputs to PCLU i.

Definition 3. 4. 7

The constants T_1 and $T_2 \epsilon\, \mathcal{P}_1$ are the lower and upper thresholds respectively.

According to the discriminant function, each mf input is multiplied by a corresponding mf weight and each feedback input is multiplied by a corresponding feedback weight. (From Figure 3. 4. 1 note that a two is equivalent to a one in this multiplication.) The total PCLU contribution is subtracted from the total mossy fiber contribution (the inhibition effect) and the result is compared with the two



(a) Schematic

$$y_i(t) = \left\lceil A_i(t) \cdot M_i(t) - B_i(t) \cdot Z_i(t) \right\rceil_{T_{12}}$$

where

$m_{ij}(t) = 1$ or 2 implies

$a_{ij}(t)\, m_{ij}(t) = a_{ij}(t)$.

(b) Discriminant function

$S_i(t)$	$m_{ij}(t) \in$	$A_i(t+d)$	$B_i(t+d)$
0	$\{0, 1\}$	$A_i(t)$	$B_i(t)$
0	$\{0, 2\}$	$A_i(t)$	$B_i(t)$
1	$\{0, 1\}$	$A_i(t)$	$B_i(t) + \delta Z_i(t)$
1	$\{0, 2\}$	$A_i(t) + \Delta M_i(t)$	$B_i(t)$

(c) Weight adjustment table

Figure 3. 4. 1. The pyramidal cell logic unit.

thresholds.

The other major part of the PCLU is the training algorithm, which adjusts the weight vector if necessary. The mode of this adjustment is determined by the septal fiber input.

Definition 3.4.8

The scalar function $s_i(t) \epsilon (0, 1)$ is the septal input to PCLU i.

Figure 3.4.1 summarizes the algorithm. Expressed verbally:

a. If $s_i(t) = 1$ and the mf input has components from the set (0, 2), then every component of the mf weight vector A having a nonzero mf input is increased by some fixed amount Δ.

b. If $s_i(t) = 0$, then no change is made in any weight vector.

c. If $s_i(t) = 1$ and the mf input has components from the set $\{0, 1\}$, then every component of the feedback vector B_i having a nonzero feedback input is increased by some fixed amount δ.

It is important to note that the connectivity of the net requires the feedback inputs Z_i to be internal inputs (see Definition 2.2.3).

3.4.3. The Basket Cell Model: The Basket Cell Logic Unit

The well-known function generator called the threshold logic unit is used as the basket cell model in the net. Renamed the basket cell logic unit, or BCLU, the representation is shown in Figure 3.4.2. The definitions of interest are:

Definition 3.4.9

$$\lceil V_i(t) \cdot H_i(t) \rceil_T = z_i'(t)$$

is the discriminant function of BCLU i, where

$$V_i \cdot H_i(t) \geq T \qquad \text{iff } z_i'(t) = 1$$

$$V_i \cdot H_i(t) < T \qquad \text{iff } z_i'(t) = 0$$

and where

$$v_{ij}h_{ij}(t) = v_{ij} \qquad \text{iff } h_{ij}(t) = 1 \text{ or } 2$$

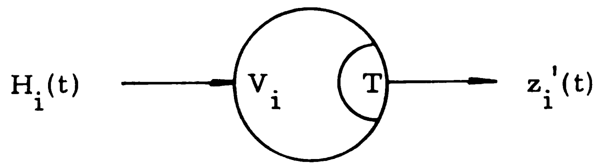
$$v_{ij}h_{ij}(t) = 0 \qquad \text{iff } h_{ij}(t) = 0 \quad \text{or} \quad v_{ij}(t) = 0.$$

Definition 3.4.10

The vector $V_i \epsilon \mathcal{P}_1$ is the vector of weights for BCLU i.

Definition 3.4.11

The positive integer T is the BCLU threshold.



(a) Schematic

$$z_i'(t) = \left\lceil V_i \cdot H_i(t) \right\rceil_T$$

where

$$v_{ij}h_{ij}(t) = v_{ij} \quad \text{iff} \quad h_{ij}(t) = 1 \text{ or } 2$$

$$v_{ij}h_{ij}(t) = 0 \quad \text{otherwise}$$

(b) Discriminant function

Figure 3. 4. 2. The basket cell logic unit (BCLU).

Definition 3. 4. 12

The vector $H_i(t)$, with components from the set $\{0, 1, 2\}$, is the vector of inputs to BCLU i.

3. 4. 4. The Connectivity and Delays

The pattern of connectivity in the CA3 sector net (Figure 3. 4. 3) is an extreme simplification of the connection scheme of the natural system. The mossy fiber input feeds a rank of PCLUs. At the output of each PCLU there is a unit delay; the output of these delays is used as the input to a rank of BCLUs and also as the output of the net. The output of each of the BCLUs first passes through a unit delay and then feeds the PCLU rank.

There are two rules that might be used when defining a specific connectivity. The first is suggested by the CA3 sector morphology: a PCLU should feed the BCLUs in only a limited surround of the PCLU, and a BCLU should feed PCLUs over an area several times as large. The second rule is suggested by the behavior of the model (as developed in the next chapter): a direct path should exist from each PCLU i to at least one BCLU and back to PCLU i. If it is assumed that only one BCLU per PCLU is connected in this fashion, then:

Definition 3. 4. 13

The BCLU in the direct PCLU i - BCLU - PCLU i path is called the special BCLU of PCLU i.

As will be seen in subsequent chapters, the extent of both the trainer and the target table generator's knowledge of a functal net's connection scheme plays an important part in determining the operating characteristics of those structures (for example, their versatility when changing the net's function). In order to emphasize this point, the connectivity of the CA3 sector net is specified only to the extent of its trainer and target table generator's knowledge. That is, it is assumed reasonable for both structures to know about the special BCLUs; it is assumed unreasonable to suppose that they know the first connectivity rule. Therefore, the special BCLU connectivity rule is the only one assumed for the CA3 sector net.

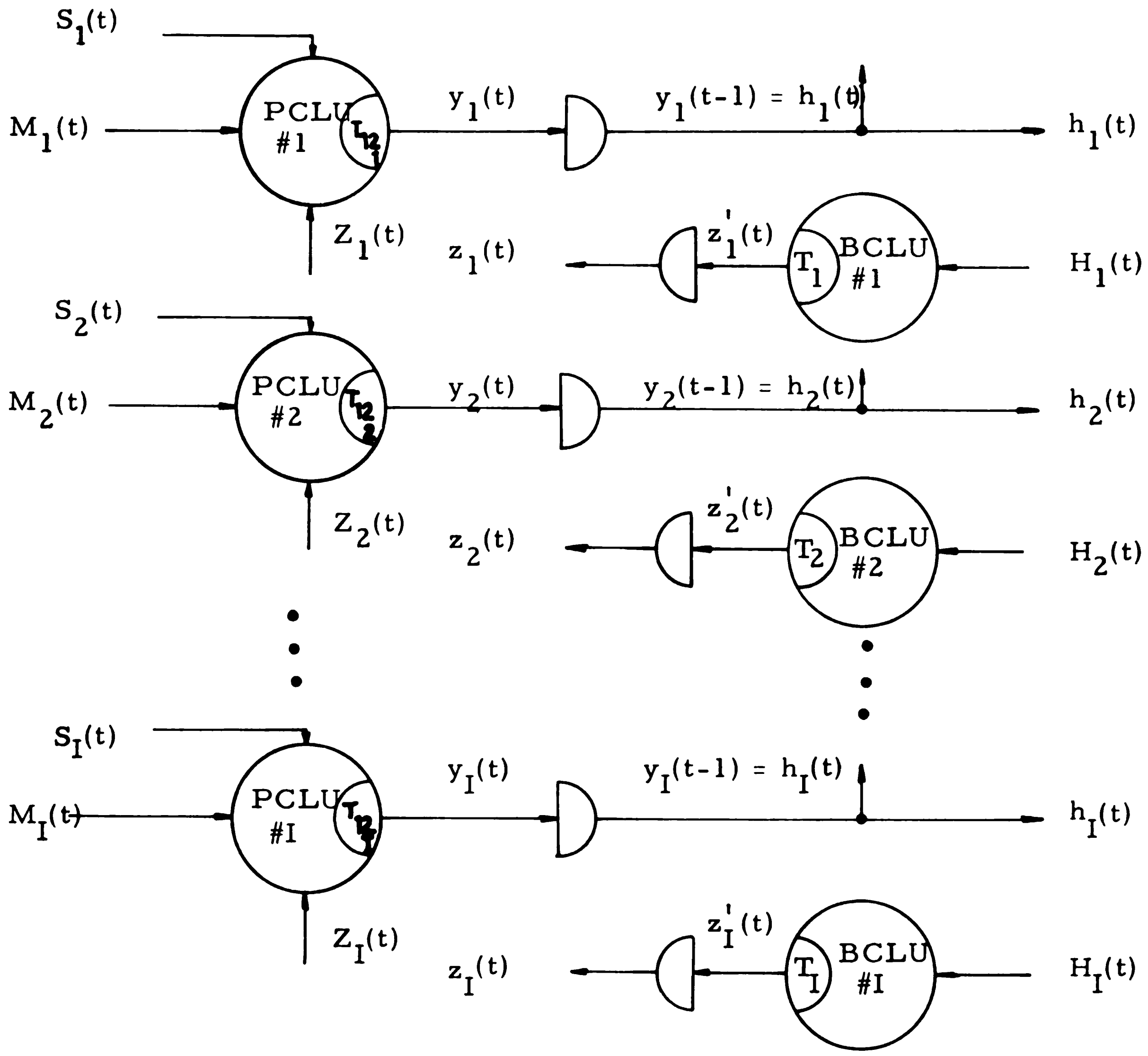


Figure 3. 4. 3. A general form of connectivity for the CA3 sector net.

A more complex and seemingly more realistic connection scheme for a CA3 sector model is presented in the Appendix. It is suggested that part of the reason for the complexity of the connectivity in the natural hippocampal system is to overcome the restraints placed on the natural trainer's activities because of its lack of knowledge of the hippocampal structure. This observation appears to present a paradox, but perhaps the explanation is that, after a certain critical level of connection complexity, more complexity tends to eliminate the need for detailed knowledge on the part of the trainer and target table generator; they can deal instead with generalities.

Finally, a comment on the delays. It may be that there has been a significant oversimplification in the placement and magnitudes of the model's delays. Unfortunately, a more complex arrangement would remove the behavior of the model from the realm of the author's existing intuition.

3.4.5. The Operational Algorithm

In order to discuss computational properties of the net it is necessary to be specific about the order in which the computations occur. This order is:

1. Advance the state.
2. Compute the new outputs of the PCLUs.
3. Compute the new weight vectors of the PCLU.
4. Compute the new outputs of the BCLUs.

3.5. The Output Buffer

The computation of the output buffer obeys the following truth table:

$h_i(t)$	$\phi(t)$
0	0
1	0
2	1

In addition, if the output buffer input CFO = 0, then all output buffer outputs are zero.

Presumably the natural structure which performs this function is the CA1 sector. This should not, however, be taken as the full extent of the functional sophistication of this area.

CHAPTER 4

PROPERTIES OF THE CA3 SECTOR NET

4. 1. Introduction

The properties of the CA3 sector net presented in this chapter are important in two ways. First, the trainer and target table generator designs depend, to a large extent, on the computational properties of the functal net they control. Second, the properties constitute an analysis of the function of recurrent inhibition as it occurs in the net.

A necessary preliminary assumption concerns the major role played by the special BCLU in net computation.

Assumption 4. 1. 1.

The output of the special BCLU of PCLU i is assumed to be nonzero whenever the input to the BCLU from PCLU i is nonzero.

The most basic CA3 sector net property is the following.

Property 4. 1. 1.

Two $M = 0$ inputs place the hippocampus net in the zero state.

Proof: Suppose the net is in some state $Q = (H, Z)$ and has a PCLU output Y and a BCLU output Z'_a. Furthermore, assume that the mossy fiber input matrix $M = 0$. The operational algorithm of the net outlined in Section 3. 4. 5 says that the next time period will see the state of the net become $Q = (Y, Z'_a)$, the PCLU output become 0 and the BCLU output become Z'_b. If the mf input remains zero for the next time period, then the state of the net, the PCLU output, and the BCLU output will become $Q = (0, Z'_b)$, 0, and 0 respectively. The state of the net for the next time period will be $Q = (0, 0)$. QED

Therefore, any time a zero state is desired it is only necessary to apply a zero input for at least two time periods.

The structure of the hippocampus and intuition made it difficult to justify the existence of the start state table, the start state table generator, and the decision components of the general functal system. By giving the change of state controller the capability to apply a zero input to the net (through the input buffer) and making the following assumption, it was possible to entirely eliminate these troublesome components from the hippocampus system model.

<u>Assumption 4. 1. 2.</u>

The only start state of a target sequence will be the zero state.

Based on this assumption, the following orthodox trainer and target table generator operating algorithm was defined.

<u>Algorithm 4. 1. 1.</u>

1. Assume the function realized by the net is also contained in the target table. Change the table to a new function which is contained in the convergence set of the original function.

2. Place the net in a zero state by applying two successive zero inputs.

3. When a conflict between the computed and the desired output of any one PCLU is detected, modify the net by increasing the mf weights if the generated output is lower in magnitude than the desired output, or the feedback weights if the generated output is higher in magnitude than the desired output of the PCLU (using the Weight Adjustment Table of Figure 3. 4. 1).

4. Reset the entire net to a zero state. Recompute the output sequence and go to Step 3 if an error is detected.

5. Training is successful when this output sequence and all others are generated correctly.

In addition, the original method of evaluating and improving the performance of this algorithm was defined to be: Maximize the intersection between each convergence set of the system and the function set of the net.

As the following example illustrates, both of these definitions proved to be unworkable because the convergence class of the net cannot be defined.

<u>Example 4. 1. 1.</u>

Let σ_k = 0 2 2 1 1 (1 1)* be an output sequence component of the function being realized by the PCLU of Figure 4. 1. 1. Suppose the corresponding target sequence component is changed to

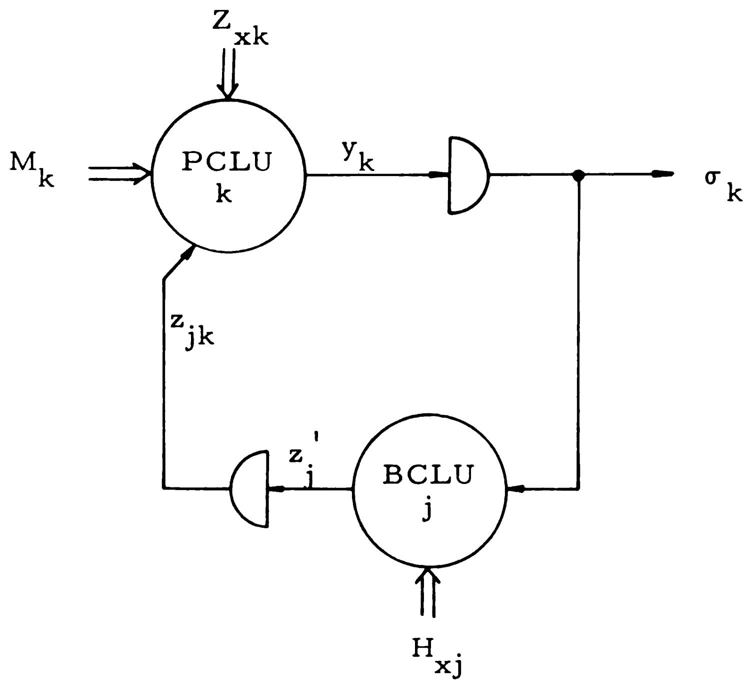


Figure 4. 1. 1. A PCLU and its special BCLU.

M_k is the mossy fiber input vector, Z_{xk} is the feedback input vector from BCLUs other than the special one, and H_{xj} is the input to the BCLU from PCLUs other than PCLU j.

$\sigma_k* = 0\ 2\ 2\ 0\ 0\ (0\ 0)*$. According to Algorithm 4. 1. 1 the feedback weights will be increased when the error at the h^2 pair position is detected. (The general sequence notation is $\sigma_k* = 0\ h^1\ h^1\ h^2\ h^2\ h^3\ h^3 \ldots$) Since the special BCLU existence is assured by Definition 3. 4. 13 and its output is assured of being nonzero whenever the input from its PCLU is nonzero, the training will succeed at least for the h^2 pair. Successful training for the next pair, h^3, cannot be guaranteed, however, since there is no assurance that either one of the following conditions is true:

1. The Z_{xk} feedback input from BCLUs other than the special BCLU are nonzero.

2. The input to the special BCLU from other PCLUs is sufficient to cause the special BCLU output to be nonzero, even though the input from PCLU k is zero.

In conclusion, it is not possible to say whether or not any function containing σ_k* is in the convergence set of any function containing σ_k.

The definition of a new algorithm and method of evaluation was based on two properties of the net discovered while evaluating Algorithm 4. 1. 1. The first is implied by the previous example: If the atom of a target table function is changed, then the atoms and elements following it in the sequence cannot be predicted. (It is important to note that this statement does not imply anything about the atoms preceding the altered atom.)

The second property involved a consideration of whether or not successful training can be guaranteed if only one atom of a target table function is allowed to change. The following example demonstrates that in some cases it would be necessary to make multiple atom changes in order to insure successful training as defined in Algorithm 4. 1. 1, Step 5.

Example 4. 1. 2.

Consider a net in state $Q = (H, Z) = 0$. The disc. fcn. of a single PCLU k is $A_k \cdot M_k - B_k \cdot Z_k^i$, which reduces to $A_k \cdot M_k$ for h^1. A well-known property of disc. fcns. of this form is: If $M_k^1 \supset M_k^2$ and $h^1(M_k^1) = a$, then $h^1(M_k^2) \geq a$. This implies that if $h^1*(M_k^1) = a$, then $h^1*(M_k^2) \geq a$. Therefore, changing one atom in the first element of a target sequence will generally require changing atoms of several other target sequences.

The problem of defining multiple atom changes is equivalent to the function set definition problem and the latter can be solved in two steps:

1. Develop an algorithm for generating target sequences.

2. Develop an algorithm for constructing the entire target table from target sequences.

It has not been possible to develop the second algorithm. Even if one was developed, however, it is unlikely that an algorithm of such apparent complexity could be imitated by a nervous system. This is especially true in light of the reasonableness of the following algorithm and method of evaluation:

Algorithm 4. 1. 2.

1. Assume the function realized by the net is also contained in the target table. Change one atom pair $h^k h^k$ in the table.

2. Place the net in a zero state by applying two successive zero inputs.

3 When a conflict between the computed and the desired output is detected, modify the offending PCLU's disc. fcn. by either (a) increasing the mf weights if the generated output is lower in magnitude than the desired output or (b) increasing the feedback weights if the generated output is greater in magnitude than the desired output.

4. Reset the net to a zero state. Recompute the output sequence and go to step 3 if an error is detected in any target sequence element through the element containing the original alteration. Otherwise, training is considered successful.

The method of evaluation and improvement was to determine the atom alteration rules which, when used during step 1, would make it possible for this algorithm to succeed.

4. 2. The Output Sequence Set of a PCLU

The output sequence set of an arbitrary PCLU in the CA3 sector net has some important properties which will ultimately allow the set of output sequences of an N PCLU net to be completely generated. The properties are also interesting in their own right.

Property 4. 2. 1.

If a net is initially in state $Q = 0$, then the output sequence of any PCLU i for any input M_i will be of the form

$$\sigma_i(M_i) = 0\, h^1\, h^1\, h^2\, h^2\, h^3\, h^3 \dots .$$

Proof: The proof can be summarized by Table 4. 2. 1,which traces the state of the net, the feedback input Z_i, and the computed output h_i^k through several time periods.

Picking up the action at t_1, the input M_i extracts an output of y^1 from the PCLU. Since the input to all the BCLUs (H) is still 0, their output is collectively 0. Therefore, the new state formed for the t_2 computations will be as shown.

For t_2 the input to the PCLUs in general and for the PCLU i in particular is no different than it was for t_1. Therefore, the output will not change. The input to the BCLUs has changed, however, and a new collective output of Z^{1}' should be expected. The state shift leaves (H^1, Z^1) as the state for the t_3 computations.

During t_3 the feedback input to the PCLUs can be nonzero for the first time. This is reflected in the change in the PCLU i output. The BCLU inputs have not changed, so their output remains the same and the output of the BCLUs is different.

It is clear that such a pattern will continue as long as the input to the net or the functions computed by the functals do not change. QED

Property 4. 2. 2.

The output sequence component $\sigma_k(M_k) = 0\, h^1\, h^1\, h^2\, h^2 \dots$ generated by PCLU k must satisfy the following set of inequalities:

(1) $h^1 \geq h^2, h^3, h^4, h^5, \dots$

(2) $h^2 \leq h^3, h^4, h^5, h^6, \dots$

(3) $h^3 \geq h^4, h^5, h^6, \dots$

(4) $h^4 \leq h^5, h^6, h^7, \dots$

(5) $h^5 \geq h^6, h^7, h^8, \dots$

$\vdots$

(Note that the k subscript has been suppressed.)

Proof: The predicate for h^1 is $\lceil A_k \cdot M_k \rceil$. The predicate for any other $j > 1$ is $\lceil A_k \cdot M_k - B_k \cdot Z_k^j \rceil$. Clearly (1) is true. Therefore,

Table 4. 2. 1.

The Computations, Over Several Periods, of the CA3 Sector Net

period	state of net $Q = (H, Z)$	feedback input Z_i	output y_i^k	delayed output h_i^k
t_o	$(0, 0)$	0	0	0
t_1	$(0, 0)$	0	y^1	0
t_2	$(H^1, 0)$	0	y^1	h^1
t_3	(H^1, Z^2)	Z_i^2	y^2	h^1
t_4	(H^2, Z^2)	Z_i^2	y^2	h^2
t_5	(H^2, Z^3)	Z_i^3	y^3	h^2
t_6	(H^3, Z^3)	Z_i^3	y^3	h^3
⋮	⋮	⋮	⋮	⋮

$HW(H^1) \geq HW(H^j)$ for all j and $H^j \supset H^1$. Since the BCLUs are threshold functions, $HW(Z^2) \geq HW(Z^j)$ and $Z^j \supset Z^2$. As a result $B \cdot Z^j \leq B \cdot Z^2$ and $h^2 \leq h^j$ for all $j \neq 2$. Therefore, $HW(H^2) \leq HW(H^j)$, $j \neq 2$. Any PCLUs which fire for h^2 will certainly fire for h^j, so $H^2 \supset H^j$. This completes the proof of (2).

Continuing, $HW(Z^3) \leq HW(Z^j)$ and $Z^3 \supset Z^j$, $j \geq 4$. Therefore, $B_k \cdot Z^3 \leq B_k \cdot Z^j$ and $h^3 \geq h^j$. $HW(H^3) \geq HW(H^j)$ and so on. QED

4. 3. The Output Sequence Set of the Hippocampus Net

The following property was implied in the proof of Property 4. 2. 1.

Property 4. 3. 1.

The CA3 sector net has output sequences of the form

$$\sigma(M) = 0\, H^1\, H^1\, H^2\, H^2\, H^3\, H^3 \ldots$$

Recall that in general the repetition of a subsequence in an output sequence does not imply that the subsequence is a cycle. The CA3 sector net is nearly an exception, but the argument for it being an exception is purely academic, as can be seen by the following property.

Property 4. 3. 2.

If $\sigma(M) = 0\, H^1\, H^1 \ldots H^i\, H^i \ldots H^j\, H^j \ldots$
and $H^i = H^j$ $(j > i)$, then $H^i\, H^i\, H^{i+1}\, H^{i+1} \ldots H^{j-1}\, H^{j-1}$
is a cycle.

Proof: Assume a state (H^{i-1}, Z^i) produced an output Y^i. This will become H^i during the next period and the new state will be (H^i, Z^i). The next state will be (H^i, Z^{i+1}). Similarly, assume a state (H^{j-1}, Z^j) produces an output Y^j. This will become H^j during the next period and the new state will be (H^j, Z^j). The next state will be (H^j, Z^{j+1}). If $H^i = H^j$, the $Z^{i+1} = Z^{j+1}$ and $(H^i, Z^{i+1}) = (H^j, Z^{j+1})$. The two equal states mark the boundaries of a cycle. QED

Properties 4. 2. 2, 4. 3. 1, and 4. 3. 2 can be combined in an algorithm which exhaustively lists all possible output sequences of a CA3 sector net containing N PCLUs.

Algorithm 4. 3. 1.

1. Generate an output H^1 from the set of 3^N possible outputs.
2. Generate an output H^2 from the same set.

3a. If the sequence $H^1H^1H^2H^2$ satisfies Property 4. 2. 2, go to 4.

3b. Otherwise, go to 2 until every possible output candidate for H^2 has been tested. Then go to 1 and repeat until every possible output candidate for H^1 has been tested.

4. If the sequence $H^1H^1H^2H^2$ satisfies Property 4. 3. 2, add the sequence to the list of output sequences and go to 3b. Otherwise K = 3 and continue.

5a. Generate an output for H^K from the set of possible outputs. If the sequence $H^1H^1H^2H^2 \ldots H^KH^K$ satisfies Property 4. 2. 2, go to 6.

5b. Otherwise, generate another output for H^K and test again until the set of outputs for the K-th element in the sequence has been exhausted. Then generate a new output for H^{K-1} and reinitialize the set of outputs to be tested for H^K. The algorithm terminates when all possible outputs for H^1 have been tested.

6. If the sequence $H^1H^1H^2H^2 \ldots H^KH^K$ satisfies Property 4. 3. 2, add the sequence to the output sequence set and go to 5b.

A version of this algorithm with the ability to generate all possible target sequences for a net with N PCLUs was programmed on the CDC 6500 computer (see Appendix B). Since it would be prohibitively expensive to allow the program to generate all possible target sequences, a representative sample was taken for several values of N and for target sequences with the first element containing all twos and the second element containing all zeros (to give target sequences of maximum length). Sixteen was the longest output run-in length found (for N=5), with the length increasing slowly with increasing N. Only output cycles of length 4 and equilibria were found; there were approximately equal numbers of each.

4.4. Rules for Successful CA3 Sector Net Training Using Algorithm 4. 1. 2.

The results presented in the previous two sections, along with those below, are sufficient to develop the rules which assure successful training using Algorithm 4. 1. 2. The key word in the following property is "guaranteed."

Property 4. 4. 1.

Using Algorithm 4. 1. 2 and its associated success criterion, training of the hippocampus net is guaranteed to be successful if and only if the following rules are obeyed. (The subscripts have been omitted for simplicity.)

Rule 1: If $\sigma(I) = 0\, h^1\, h^1\, \sigma'(I)$, then changes in h^1 are made according to the following table.

h^1	$h^{1}*$
0	1
0	2
1	2

provided $\Delta < \frac{T_2 - T_1}{N}$, N the number of mf inputs to PCLU j

Rule 2: If $\sigma(I) = 0\, h^1\, h^1 h^2 h^2 \sigma'(I)$, then changes in h^2 are made according to the following table

h^1	h^2	$h^{2}*$
2	0	1
2	0	2
2	1	2
2	2	0
2	2	1
2	1	0
1	1	0

provided $\Delta < \frac{T_2 - T_1}{N}$, N the number of mf inputs to PCLU j

provided $\delta < \frac{T_2 - T_1}{L}$, L the number of feedback inputs to PCLU i.

Rule 3: If $\sigma(I) = 0(2200)(2200)*\, h^i h^i\, \sigma'(I)$ and $h^i = 1$, then $h^{i}* = 0$ or 2.

Rule 4: If $\sigma(I) = 022(0022)*\, h^i h^i\, \sigma'(I)$ and $h^i = 1$, then $h^{i}* = 0$ or 2.

Proof:

"Rule 1: Assume the net is realizing the function in the target table; change the atoms h^{1*}. Clearly, if h^{1*} is increased to 2, then h^1 can be increased to 2 by increasing the mf weights and training will be successful. If h^{1*} is increased to 1, it is necessary that an increase in the mf weights not force an output of $h^1 = 2$. The condition $\Delta < \frac{T_2 - T_1}{N}$ will prevent this from happening, since any one increment of the disc. fcn. will be less than the $T_2 - T_1$ gap. Note that h^1 can never be decreased, since the feedback input for h^1 will always be zero vector. "

Rule 2: The table associated with Rule 2 defines the changes that can be made in the second pair of atoms of a target sequence component with guaranteed success. The changes in h^{2*} are dependent on h^1, since this output element defines the upper bound on any change. If h^2 must be increased from 0 to 1, then the same Δ limit must be observed as was defined in the proof of Rule 1. There is, of course, no problem if h^2 is increased to 2 (assuming $h^1 = 2$). But note that the alternative $h^1 = 1$ and h^2 is increased from 0 to 1 has been omitted from the table. Any attempt to increase the disc. fcn. to produce $h^2 = 1$ under these conditions may inadvertently produce $h^1 = 2$. Since h^1 cannot be decreased, the training would have to be considered a failure.

In general, H^2 is the first output element associated with a nonzero feedback input. The existence of the special BCLU guarantees that if h^1 is nonzero, the feedback input vector is nonzero. This in turn guarantees that the disc. fcn. of the PCLU j can be decreased by increasing the feedback weights. This is the justification for the inclusion of the last four entries in the table under Rule 2. Note that a change in h^2 from 2 to 1 requires a condition on δ. This condition prevents the disc. fcn. from dropping from a value above T_2 to a value T_1 or lower with a single increment of the feedback weights.

Rule 3: This rule summarizes the changes that can occur with guaranteed success when the atom altered is $h^{i}*$, $i \geq 3$ and odd. Suppose $h^{i}*$ is increased. From Property 4. 2. 2, the bound on this increase is determined by h^{i-2}. The possible changes are:

	h^{i-2}	h^i	$h^{i}*$
(a)	1	0	1
(b)	2	0	1
(c)	2	0	2
(d)	2	1	2

For alternatives (a), (b), and (c) $h^i = 0$ implies $h^k = 0$, all even $k < i$ (using Property 4. 2. 2). If any of these changes are made, then, when the error in the output is detected, the reaction of the trainer is to increase the mf weights. In doing so, it is entirely possible that some of the disc. fcns. of the even elements will be inadvertently increased over the T_1 threshold. When the PCLU generates the incorrect output sequence element upon reinitialization, the response of the trainer is to increase the feedback weights. The possibility exists that this will force the disc. fcn. of h^i to fall below the desired threshold. To correct this, the mf weights are increased again, creating the situation where the even elements may again become incorrect. The trend is clear and the conclusion is that success cannot be guaranteed if any of changes (a), (b), or (c) are made.

From the information given and Property 4. 2. 2, the output sequence component associated with alternative (d) is of the form

$$\sigma(I) = 0\ (2\ 2\ h^2\ h^2)(2\ 2\ h^k\ h^k)*\ \underbrace{1\ \ 1}_{h^i\ h^i}\ \sigma'(i)$$

where $k < i$ and even, h^2 and $h^k \epsilon\ (0, 1)$ and Property 4. 2. 2 holds. If $h^k = 1$ for any even $k < i$, then the situation is the same as in the other three alternatives: there is the possibility of unstable training. Therefore, all sequence components are eliminated except those of the form suggested by the rule itself. The crucial step in the proof is to demonstrate that the fatal trainer instability of the other alternatives does not occur.

Let the j-th PCLU generate $\sigma_j(I)$ and change h^i* to 2. When the change is first detected by the trainer, the mf weights are increased to produce the correct value of the disc. fcn. for h^i, $D^i(t_1) \geq T_2$. However, as in the previous alternatives, $D^k(t_1) \geq T_1$ may be true for some even $k < i$. In order to compensate for this error, the trainer increases the feedback weights, thus decreasing the disc. fcns. until, in particular, $D^k(t_2) < T_1$. So far the script is the same as in all of the other alternatives. Note, however, that originally $h^k < h^i$, $k < i$ and even. Since $Z_j^i \supset Z_j^k$, $k < i$ and even, $h^i > h^k$ implies $HW(Z_j^i) < HW(Z_j^k)$. The important property is the strictly less than of the Hamming weight relation. This implies that the change in the disc. fcn. for h^i is strictly less than the change in the disc. fcn. for h^k:

$$D^i(t_1) - D^i(t_2) < D^k(t_1) - D^k(t_2)\ .$$

If $D^i(t_2) < T_2$, the trainer will attempt to compensate by increasing the mf weights again. This time, if conditions are right,‡ $D^k(t_3)$ will be less than $D^k(t_1)$. If $D^k(t_3)$ is still greater than T_1, the compensation in the feedback weights need be no greater than the compensation for $D^k(t_2)$, and it can be less. If $D^i(t_4)$ is still less than T_2, the mf weights will be increased less than the increase that occurred during the computation of $D^i(t_3)$. Eventually $D^i(t_n) \geq T_2$ while at the same time $D^k(t_n)$ is not increased enough to force h^k to be incorrect.

To complete the proof, note that any changes in the k-th component, $k \leq i$, are corrected before the change can affect the other PCLUs. Therefore, the other sequence components through target sequence element i do not change during the training for the k-th component.

Now suppose h^i, $i \geq 3$ and odd, is decreased. The bound on the decrease is determined by h^{i-1} and the possible changes are (again from Property 4. 2. 2):

	h^{i-1}	h^i	h^i*
(a)	0	1	0
(b)	0	2	1
(c)	0	2	0
(d)	1	2	1

Alternatives (b), (c), and (d) can be eliminated in short order as successful training candidates. In all cases $h^k = 2$, $k < i$ and odd,

‡ Since the mf weights increase in "quantum jumps," it would, in general, not be possible to recompute $D^i(t_1)$ exactly; the value actually computed may range from a quantum higher to a quantum lower. If it is the former, then it is possible that the difference $D^i(t_3) - D^i(t_2) \geq D^i(t_1) - D^i(t_2)$. In this case, the feedback weights would be required to increase the same amount as before to correct h^k. However, the next time the mf weights are increased, the increment required for a correct h^i will be even less than before. Eventually this extra negative weight will be great enough that the contribution of the mf weights will be less than the contribution of the feedback weights, no matter what the magnitude of the quantums, and the proof will proceed as outlined.

and it is entirely possible that $Z_j^k = Z_j^i$ for at least one of those k's. If this is so, then any attempt to decrease D^i by increasing the feedback weights decreases D^k by the same amount. Therefore, h^k will become incorrect at the same time h^i becomes correct. The trainer will respond by increasing the mf weights, but the effect is felt equally by both h^k and h^i. The result is training instability.

Alternative (a) implies output sequence components of the form

$$\sigma(I) = 0\,(h^1\ h^1\ 0\ 0)(h^k\ h^k\ 0\ 0)*\ \underbrace{1\ \ 1}_{h^i\ h^i}\ \sigma'(I)$$

where $k < i$ and odd and h^1, $h^k \epsilon$ (1, 2), along with Property 4.2.2. If any of the h^k or h^1 is one, then training instability may occur. This leaves only output sequences of the form given in the rule statement. In order to reduce h^i, the feedback weights are increased, and all of the disc. fcns. D^k, $k < i$ and odd, are reduced. Perhaps some will be reduced to below T_2. Consequently, the mf weights will be increased to compensate, with the possibility that D^i is forced to a value above T_1. Fortunately, a property of the same nature as described in alternative (d) of the previous set exists to prevent training instability: Since $Z_j^k \supset Z_j^i$, if $h^i < h^k$ for all k originally, then $HW(Z_j^i) > HW(Z_j^k)$. Therefore, the D^k will not be decreased as much as D^i, and eventually $D^i < T_1$, while $D^k \geq T_2$ for all k.

Rule 4: The final rule summarizes the changes that can occur with guaranteed success when the atom altered is h^i, $i \geq 2$ and even. If h^i is increased, then the upper bound is determined by h^{i-1} and the possible changes are:

	h^{i-1}	h^i	h^i*
(a)	1	0	1
(b)	2	0	1
(c)	2	0	2
(d)	2	1	2

Successful training for the (a), (b), and (c) alternative cannot be guaranteed, since $h^i = 0$ implies that $h^k = 0$, $k < i$ and even.

The output sequence components accompanying alternative (d) are of the form:

$$\sigma(I) = 0\ 2\ 2\ (h^k\ h^k\ 2\ 2)^*\ \underbrace{1\ 1}_{h^i\ h^i}\ \sigma'(I)$$

where k is even, h^k, $h^i \epsilon (0, 1)$. The subset of sequence components where $h^k = 1$ for any k can be immediately eliminated, leaving sequences of the form given in the rule. Successful training is guaranteed for these by the same argument as was used for (d) in the first set of alternatives in Rule 3.

If h^i is decreased, then the lower bound will be determined by h^{i-2} and the possible changes are:

	h^{i-2}	h^i	h^{i*}
(a)	0	1	0
(b)	0	2	1
(c)	0	2	0
(d)	1	2	1

Successful training for alternative (b), (c), and (d) cannot be guaranteed since $h^i = 2$ implies $h^k = 2$, $k <$ i and odd.

The output sequence components accompanying alternative (a) are of the form:

$$\sigma(I) = 0\ 2\ 2\ (h^k\ h^k\ 2\ 2)^*\ \underbrace{1\ 1}_{h^i\ h^i}\ \sigma'(I)$$

where k is even and h^k, $h^i \epsilon (1, 2)$. Again the subset of sequence components where $h^k = 1$ for any k can be eliminated, leaving sequences of the form given in the rule. Successful training is guaranteed for the remainder by the same argument as was used for (a) of the second set of alternatives in Rule 3. QED

CHAPTER 5

A FUNCTAL SYSTEM MODEL OF THE HIPPOCAMPUS. PART 2.

5. 1. The Target Table

The previous chapter noted that if a net is realizing the function in the target table and then one pair of atoms is changed, the output function of the net after training could then differ greatly from the function in the table. Consequently, if orthodox functal system training techniques were used, that is, if the entire new target table had to be realized by the net, training would be unstable and the net would be essentially useless. The following assumption summarizes a target table form different from the one originally defined in Section 2. 7 which helps to circumvent this problem.

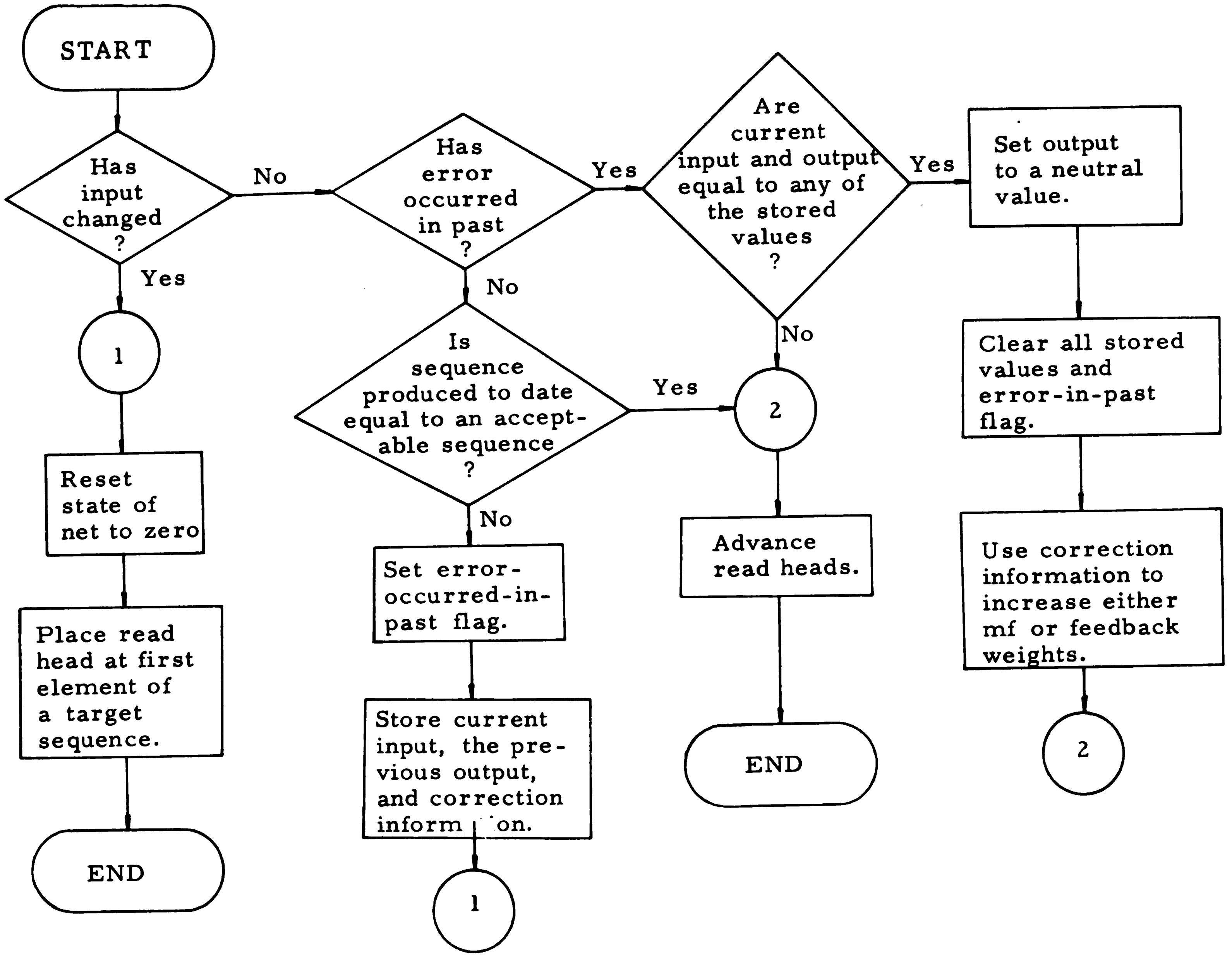
Assumption 5. 1. 1.

The target table can contain a set of target sequences for each input. The interpretation given to each set of target sequences is: Any output sequence not contained in a set for a particular input is considered to be harmful to the entity of which the hippocampus or its model is a part. Those output sequences which are target sequences are either neutral or beneficial to the entity.

The target table is a conceptual device which makes explicit the relationship between the natural system and its environment. It is not intended that a physical structure exist to hold the table. All neuroscientific interpretations of target tables must comply with this fact.

5. 2. The Trainer

Algorithm 4. 1. 2. has been modified to be compatible with the new target table concept. The new trainer operating algorithm for one time period is outlined in Algorithm 5. 2. 1 and the trainer structure associated with it is given in Figure 5. 2. 1. The following is a description of the algorithm.


START
Has input changed ?
No
Yes
1
Reset state of net to zero
Place read head at first element of a target sequence.
END
Has error occurred in past ?
Yes
No
Is sequence produced to date equal to an acceptable sequence ?
Yes
No
Set error-occurred-in-past flag.
Store current input, the previous output, and correction information.
1
Are current input and output equal to any of the stored values ?
Yes
No
2
Advance read heads.
END
Set output to a neutral value.
Clear all stored values and error-in-past flag.
Use correction information to increase either mf or feedback weights.
2

Algorithm 5. 2. 1. The trainer operating algorithm.

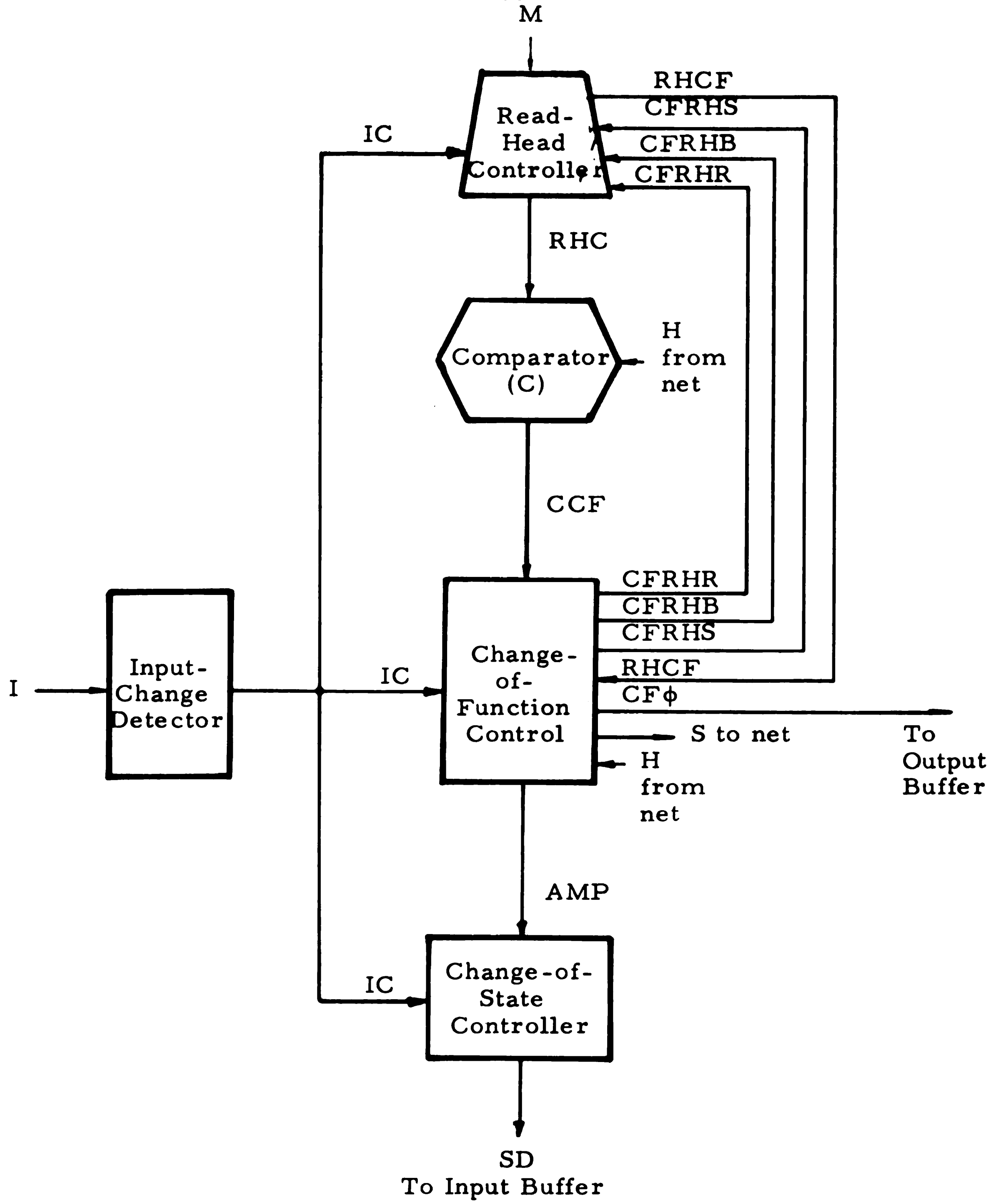


Figure 5. 2. 1. The training structure of the functal system model of the hippocampus.

At the beginning of the time period, the input-change detector examines the input for any change since the last time period. If a change occurred, then IC = 1 and the trainer does the following:

1. Its change-of-state controller sets SD = 0 for two time pulses. This holds the mossy fiber inputs at zero for the same length of time. (See Table 5. 2. 1.)

Table 5. 2. 1.

The Truth Table for the Change-of-State Controller

IC	AMP	SD
0	0	1
0	1	2
1	0	0
1	1	0

2. Its read-head controller positions the read head at the beginning of the appropriate target sequence tape.
3. Its change-of-function controller ignores the comparator output.

If IC = 0, then the change-of-function controller interrogates the flag ERROR to determine if an uncorrected error has occurred since the current input was applied.

If ERROR = 0, then an error has not occurred and the comparator compares the current output with the target sequence element information on line RHC (see Section 5. 4.). If there is a match, then the output of the comparator, CCF = 1. The change-of-function controller senses this and sends the read-head controller a signal CFRHR = 1 for one time pulse in order to advance the read-head. If there is no match, then the change-of-function controller outputs CFRHS = 1 for a time pulse. This causes the read-head controller to search through the other target sequences in the set assigned to the current input for another target sequence candidate. If one is not found, then the change-of-function controller is notified by RHCF = 1. In response, it sets the flag ERROR = 1 and stores the current input, the previous output, and the error correction information‡ in a memory area called STORE; if STORE is full, the error is ignored.

If ERROR = 1, then the current input and output are compared with each of the sets of data in STORE. If an identical pair is not found, then CFRHR = 1 for one time pulse to advance the read-head. If an identical pair is found, then the change-of-function controller sets the output of the output buffer to a neutral value (CFO = 1) for the next time period‡ ‡, clears all the stored values‡, sets ERROR = 0, and uses the error correction information to set AMP or SD. If the AMP is set, indicating an mf weight adjustment, then the change-of-state controller computes SD according to Table 5. 2. 1. Finally, the read-head is advanced with a CFRHR = 1 signal.

5. 3. The Error Correction Information for the Change of Function Controller

As described in the previous section, when a bona fide error is detected by the change-of-function controller, error correction information is stored as well as the current input and the output preceeding the erroneous one. This information consists of a PCLU

‡ Any error procedure will require that the net be in the state that led to the error. From the results of Chapter 4, the output just prior to the erroneous output is uniquely related to that state. Therefore, if the weights are activated exactly when this output is detected, the state for the next period will be the one desired. The rules for determining which weights are adjusted, i. e., the error correction information, are discussed in the next section.

‡ ‡ Note that the weights are adjusted after the output is computed. Therefore, the output will be the same erroneous value it was before, even though the weights are changed. The CFO = 1 signal prevents this output from forcing the same erroneous, and presumed dangerous, response from the entity containing the model, thus increasing the chances for survival.

‡ If the stored values are not cleared, then the problem of training interference exists. This condition occurs when the training for one target sequence unintentionally changes the state sequence of another erroneous sequence. If this happens, then the output stored for that sequence may never occur, or it might occur in the wrong place in the sequence. This is another technique to increase the chances for survival.

number and the weight vector, mf or feedback, to be adjusted. These two quantities can be determined by a number of different methods. Among them:

Method 1: Randomly select both the PCLU and the weight vector to be modified.

Method 2: Select the PCLU whose disc. fcn. is closest to one of the thresholds. In a sense this PCLU is the most uncertain of its output components. A weight vector is chosen which will force the disc. fcn. toward the nearest threshold.

Method 3: Compare the computed output with the target-sequence element and determine the erroneous PCLU. A weight vector to be modified is chosen which will adjust the disc. fcn. in the right direction.

There are advantages and disadvantages to each of these methods. Method 1 should be the slowest to arrive at an acceptable output, a possible disadvantage, but the change-of-function controller does not need to know the CA3 sector net output, a possible advantage. Method 2 is perhaps the most "neurophysiological" method. The change-of-function controller would require the values of the disc. fcns. of each of the PCLUs. This might be a disadvantage in any model, but such information could be easily provided in the natural system if the firing rate were proportional to the disc. fcn. over the range of interest. The change-of-function controller would also require some idea of the average threshold values of the PCLUs. In the natural system this information could be genetically provided.

Method 3 is the most direct method, and the one best adapted to simulation. However, the change-of-function controller would require the target sequence elements and the hippocampus output as inputs. It can be reasonably argued that if the target sequence elements are actually stored somewhere else, as this method implies, then the existence of the hippocampus can not be justified, since the storage of sequences is its proposed main function.

5.4. The Read/Write Head and Its Controller

The read/write head and its controller are organized in a manner similar to that described in Example 2.9.1. The target table is assumed to consist of a number of multidimensional tapes, one tape for each target sequence (see Figure 5.4.1). The dimension of the tapes is exactly $N + 1$, where N is the number of PCLUs in the net: The extra dimension is for the beginning-of-cycle indicator.

The tapes are read by a read-head, which is under the control of the read-head controller. The exact position of the read-head is located by the triple (M, T, E), where M is the mf input to the net, T is a number indicating the target sequence within the set for the input, and E is the number of an element within the sequence. The various inputs to the read-head controller from the other elements of the hippocampus system model, and their effect on this triple, are:

CFRHB: An input from the change-of-function controller of the trainer. Sets $E = 1$.

CFRHR: Also an input from the change-of-function controller. This sets $E = E + 1$ unless E is equal to the far right element of the sequence. If this is true, the controller uses the beginning of cycle dimension on the tape to reposition the read-head to the correct element in the sequence.

IC: An input from the input change detector. Sets $E = 1$ and $T = 1$.

CFRHS: An input from the change-of-function controller. If this input is one, $T = T + 1$, unless every target sequence in the set for this input has been inspected, in which case $T = 1$ and $RHCF = 1$. After T is increased, the new tape is inspected to determine if all elements prior to the element indicated by E are equal to the corresponding elements of the old tape. If they are not, then T is increased again.

The outputs of the read-head controller are RHCF and RHC, the latter being the contents of the element currently under the read-head.

The target table generator modifies the target table through the write-head controller. The details of operation of this controller do not concern us here.

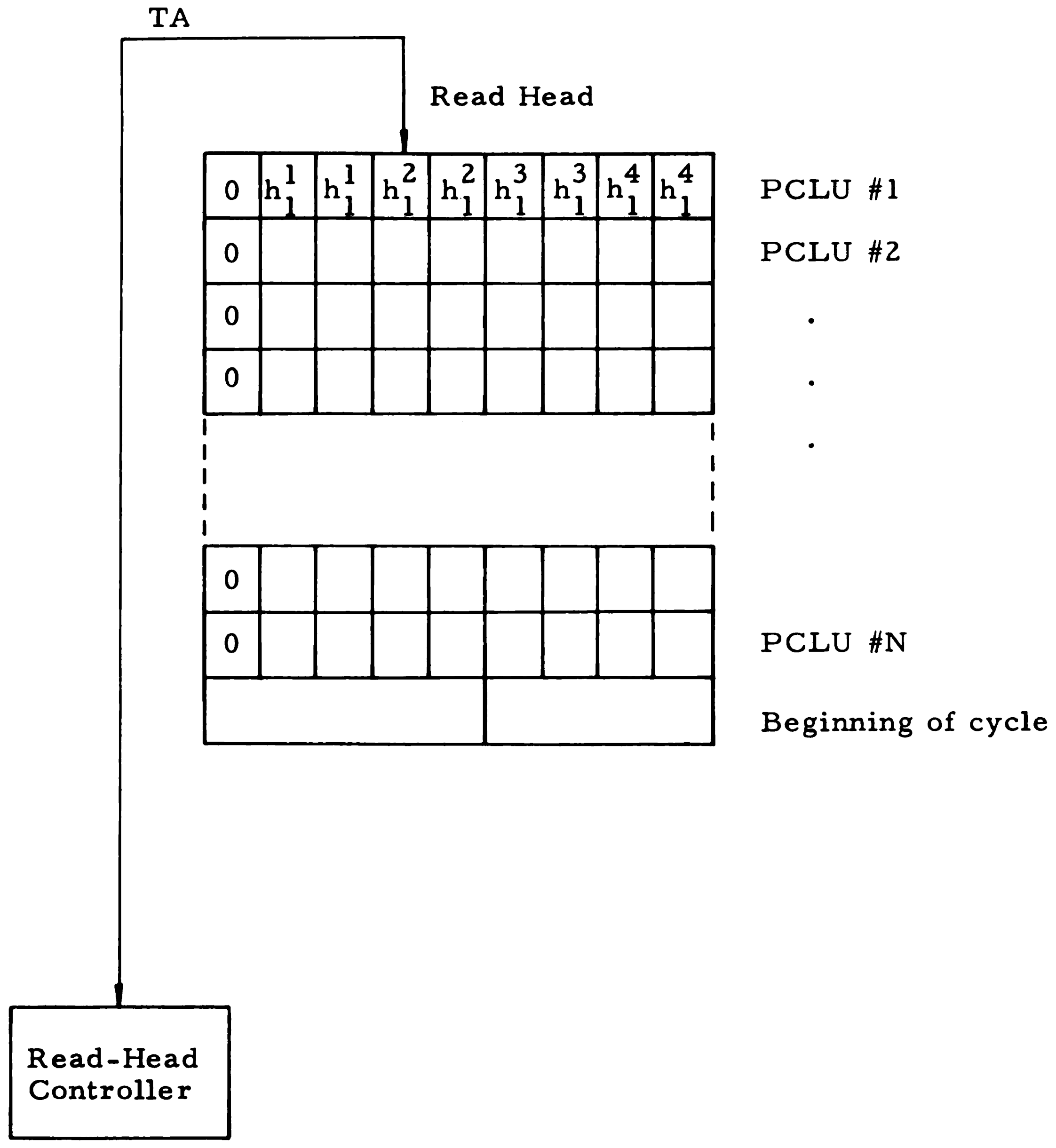


Figure 5. 4. 1. Read-head controller, target table relationship for a target table sequence.

CHAPTER 6

THE INITIAL CONDITIONS PROBLEM

6. 1. The Phase Concept

The behavior of the entity of which the hippocampus system model is a part is assumed to be influenced by the output of the CA3 sector net. Furthermore, in any steady state situation, the function generated by the net is the same as the function in the target table. Any changes in the target table are made, of course, by the target table generator. Presumably, it receives a signal from a performance evaluator whenever the response of the environment to past behavior is considered harmful. Upon receipt of such a signal, the generator makes a change in an atom of the target table. In doing so, it consults the rules developed in Chapter 4 and perhaps other rules on how to make the response more palatable to the judgment device.

A tacit assumption in this description is that the target table contains a function at the beginning of the period of interest. The problem of the initial assignment of the functions to the net and target table is not trivial: the assignment should be realistic in terms of hippocampus data and it should insure maximum survivability to the entity of which it is a part. Another problem that cannot be overlooked is the placement of the initial function in the net.

Fortunately, a bit of hippocampus data offers a plausible solution to both problems. Purpura [15] has observed that basket cells do not make connections to the pyramidal cell somata until about the third week after birth in kittens. This suggests that at birth the output of the hippocampus is generated by pyramidal cells that are not yet communicating to each other through basket cells. Furthermore, it must be assumed that the function in the hippocampus at birth is something other than the trivial function, and that the function, in part, controls the newborn kitten's instinctive behavior. It is also reasonable to extrapolate on these data and to assume that when the basket cell axons do begin to make connections, they are at first limited to the pyramids immediately surrounding the basket cell.

In applying these ideas to the solution of the two aforementioned problems, it seemed reasonable that if nature felt maximum survivability was insured by a hippocampus initially containing no recurrent inhibition, then the initial function of the CA3 sector net need not be any more sophisticated than what can be generated by a CA3 sector net with no recurrent inhibition. This and another idea on the subsequent growth of the basket cells were formalized by assuming that the lifetime of the net is divided into the following three phases:

Phase 1 -- The functions to be generated by the net are from the set of functions that can be generated by nets not having recurrent inhibition.

Phase 2 -- The functions to be generated by the net are from the set of functions which can be generated by a net with only PCLU-to-special BCLU-to-PCLU connections.

Phase 3 -- No restrictions.

6.2. Phase 1: Target Table Training

It should be clear that the target sequences generated during Phase 1 will be equilibria with components taken from the set $\{00*, 01*, 02*\}$. To illuminate some secondary principles, assume that the only target sequence components allowed during Phase 1 are‡ $\{00*, 02*\}$. Then the initial function assignment is reduced to one of defining the relationship between the entity's behavior and all of the net's 2^N output vectors with components from the set $\{0, 2\}$. This problem, although considerably simpler than the general initial function assignment problem, is still formidable. When a function of the right form is found, however, the results developed in this

‡ Note that this is comparable to saying that the pyramidal cell output is either firing at a very high rate or a very low rate, but not at a moderate, or "fine-tuned" rate.

section permit it to be placed in the target table and trained into the net prior to the net's placement in its environment.

The first result is an algorithm for generating any function that can be generated by the net while it is in Phase 1. It is based on a property presented informally in the second example of Section 4. 1 and more formally here:

Property 6. 2. 1

Let M_k^1 and M_k^2 be two different mf inputs to a PCLU k. If

$$M_k^1 \subset M_k^2 \quad \text{and} \quad h_k^1 (M_k^2) = a,$$

then $h_k^1(M_k^1) \geq a$. The algorithm itself simply assigns function values arbitrarily to all inputs with Hamming weight of one, and then uses the above property to generate restrictions on the assignment of function values for all inputs with Hamming weight of two. Successive application of the property for inputs with successively larger Hamming weights will result in a complete function;

Algorithm 6. 2. 1

Do Steps 1-5 for each PCLU in the net. ‡

1. Initialize by assigning values to h^1 for all M s. t. $HW(M) = 1$.

2. For each $h^1(M) = 2$ assigned, replace all zeros in M with x. Call the resulting set μ, with elements M^*.

RULE: It must be true that $h^1(M^*) = 2$.

3. Consider all inputs M s. t. HW(M) is one greater than the last Hamming weight considered. Assign values to the h^1 for these M within the restrictions set by RULE.

4. If the current Hamming weight is N, the dimension of the mf input vector, then go to 5; otherwise go to 2.

5. The output sequence components are of the form $\sigma(M) = 0\, h^1*$.

Example 6. 2. 1

Let $N = 5$ for the PCLU under consideration.

‡ The "k" subscript (for PCLU k) has been omitted from the description.

Step 1. Assume it is desirable to assign values from the range set (0, 2) to the mf inputs with a Hamming weight of 1 in the manner indicated by column A of Table 6. 2. 1.

Step 2. The set μ^* is (xxxx1, xxx1x). This set automatically assigns a range value of 2 as shown in column B of Table 6. 2. 1.

Step 3. The inputs with a Hamming weight of 2 not yet assigned range values are 01100, 10100, and 11000. The values chosen are as indicated in column C of Table 6. 2. 1.

Step 4. N = 2. Therefore, go to 2.

Step 2. The set μ^* contains the additional elements 1x1xx, 11xxx, and x11xx. An assignment of 2 is forced on the only input not yet paired with a range value, as is shown in column D. The algorithm terminates at this point because the function is complete.

The following example demonstrates a disturbing property of the hippocampus system model: It is not possible to train the net to realize every function the net is theoretically capable of generating during Phase 1.

<u>Example 6. 2. 2</u>

Suppose a portion of the target table for one PCLU k is:

$$h^1(M_k^1 = 00011) = 2$$

$$h^1(M_k^2 = 00101) = 0$$

$$h^1(M_k^3 = 01001) = 2$$

$$h^1(M_k^4 = 10001) = 2$$

$$h^1(M_k^5 = 10100) = 2$$

Assume W = 0 initially and the inputs are presented in the order of their superscripts. At the completion of training for M_k^1,

$$A = (0, 0, 0, a\Delta, a\Delta),$$

where a is an integer, Δ is the mf weight increment, and $2a\Delta \geq T_2$. If a and Δ are in the right proportion, then at the completion of training for M_k^3,

$$A = (0, \frac{1}{2}, 0, 1, \frac{1}{2} \cdot 3)\, a\Delta.$$

Table 6. 2. 1.

The Function Assignment for Example 6. 2. 1.

mf input	A	B	C	D
0 0 0 0 0	0			
0 0 0 0 0	2			
0 0 0 1 0	2			
0 0 0 1 1		2		
0 0 1 0 0	0			
0 0 1 0 1		2		
0 0 1 1 0		2		
0 0 1 1 1		2		
0 1 0 0 0	0			
0 1 0 0 1		2		
0 1 0 1 0		2		
0 1 0 1 1		2		
0 1 1 0 0			2	
0 1 1 0 1		2		
0 1 1 1 0		2		
0 1 1 1 1		2		
1 0 0 0 0	0			
1 0 0 0 1		2		
1 0 0 1 0		2		
1 0 0 1 1		2		
1 0 1 0 0			0	
1 0 1 0 1		2		
1 0 1 1 0		2		
1 0 1 1 1		2		
1 1 0 0 0			0	
1 1 0 0 1		2		
1 1 0 1 0		2		
1 1 0 1 1		2		
1 1 1 0 0				2
1 1 1 0 1		2		
1 1 1 1 0		2		
1 1 1 1 1		2		

At the completion of training for M_k^4,

$$A = (\frac{1}{4}, \frac{1}{2}, 0, 1, \frac{7}{4})\ a\Delta .$$

Finally, when the training for M_k^5 is complete,

$$A = (\frac{5}{8}, \frac{1}{2}, \frac{3}{8}, 1, \frac{7}{4})\ a\Delta .$$

Note that $h^1(M_k^2) = 2$, which is incorrect. It is not possible for the trainer to correct this error. Therefore, the function in the target table will not be realized by the net.

Fortunately, an algorithm has been developed which generates trainable Phase 1 functions. The algorithm generates the function assigned to only one PCLU, but since the PCLUs remain independent throughout Phase 1 (the feedback weight vector remains 0), it can be applied to each PCLU to generate the entire function. A required definition is that the set of mf inputs in the domain of h^1 which have not yet been given values in the range is called the <u>pool</u>. The initial pool, containing 2^N-1 elements, is denoted by $\mathcal{P}_1$.

<u>Algorithm 6. 2. 2</u>

1. Let $K = 1$ and $X_k = 0$.

2. $\mathcal{S}_k = \{M : HW(M) = K \text{ and } M \epsilon \mathcal{P}_k\}$. Assign $h^1(M)$, $M \epsilon \mathcal{S}_k$, as desired except that the following rule must be obeyed.

RULE: If $h^1(M) = 2$, $M \epsilon \mathcal{S}_k$, then $M \subset X_k$.

3. $K = K + 1$. $\mathcal{P}_k = \mathcal{P}_{k-1} - \mathcal{S}_{k-1} - \mathcal{R}_{k-1}$, where

$$\mathcal{R}_J = \{M : HW(M) > J, \quad M \subset M', \quad M' \epsilon\ \mathcal{S}_J \text{ and } h^1(M') = 2\} .$$

If $\mathcal{P}_k = \phi$, continue. Otherwise, select X_k from the set:

$$\{M : HW(M) = K-1, \quad h^1(M) = 0, \text{ and } M \subset X_{k-1}\}$$

and go to 2.

4. For each M, $h^i(M) = h^1(M)$ $i > 1$. STOP.

<u>Example 6. 2. 3</u>

Let $N = 6$.

Step 2. $\mathcal{S}_1 = \{000001, 000010, 000100, 001000, 010000, 100000\}$.

RULE is inconsequential for this set. Suppose all members of $\mathcal{S}_1$ are assigned a 0 output.

Step 3. K = 2, $\mathcal{P}_2 = \mathcal{P}_1 - \mathcal{S}_1$, $\mathcal{R}_1 = \phi$. $\mathcal{P}_2$ is not empty. Let $X_2 = 000001$.

Step 2. $\mathcal{S}_2$ has 15 elements. Only the elements in the set $\{100001, 010001, 001001, 000101, 000011\}$ satisfy RULE. Suppose $h^1(000011) = 2$; the remainder are assigned 0 outputs.

Step 3. K = 3. $\mathcal{P}_3 = \mathcal{P}_2 - \mathcal{S}_2 - \mathcal{R}_2$. $\mathcal{P}_2$ is not empty. Let $X_3 = 000101$.

Step 2. Only the elements in the set $\{100101, 010101, 001101\}$ satisfy RULE. Suppose $h^1(001101) = 2$; the remainder are assigned 0 outputs.

Step 3. K = 4. $\mathcal{P}_4 = \mathcal{P}_3 - \mathcal{S}_3 - \mathcal{R}_3$. $\mathcal{P}_3$ is not empty. Let $X_4 = 010101$.

Step 2: $\mathcal{S}_4 = \{110101\}$. This is also the only element which satisfied RULE. Suppose $h^1(110101) = 2$.

Step 3. K = 5. $\mathcal{P}_5 = \mathcal{P}_4 - \mathcal{S}_4 - \mathcal{R}_4 = \phi$. STOP.

The ability of the system to train the net to realize an "Algorithm 6. 2. 2" function depends on the following conditions being satisfied.

Conditions 6. 2. 1

1. The net is initially generating the trivial function, with all W = 0.

2. The function generated by Algorithm 6. 2. 2 is in the target table.

3. The mf input vectors are presented to the net in order of increasing Hamming weight.

4. Each input is held for as long as is required to train the net to generate the correct output.

In addition, there is a fifth condition consisting of two relations between the values assigned to Δ, T_1, and T_2, that requires a more lengthy discussion. One of these, relating Δ and T_1, is particularly complex, and the following property is presented in an attempt to ease the shock of the more general result. Note that the sets $\mathcal{S}_k$ defined in Algorithm 6.2.2 are required, but since they must be computed anyway, this is not an inconvenience. Also, once training is complete for the inputs in ^2_j, no other inputs will require a training session.

Property 6.2.2

For every PCLU in the net, if

(a) Conditions 6.2.2 are satisfied,

(b) $J^{N-J+1}\Delta \geq T_2$, where J is the lowest K for which $\mathcal{S}^2_K \neq \phi$,

$$\mathcal{S}^2_K = \{ M : M \in \mathcal{S}_K \text{ and } h^1(M) = 2 \} ,$$

and N is the dimension of the mf input to the PCLU.

(c) $(J-1)\, J^{N-J} \sum_{i=1}^{N-J+1} J^{1-i} < T_1/\Delta$.

(d) $|\mathcal{S}^2_J| = N - J+1$,

then Algorithm 5.2.1 will successfully train the net to realize the function in the target table.

Proof:

Let

$$\mu^2_K = \{ M : HW(M) = K \text{ and } h^1(M) = 2 \} ,$$

$$\mu^o_K = \{ M : HW(M) = K \text{ and } h^1(M) = 0 \} ,$$

$$\mu^o = \bigcup_{i=1}^{N} \mu^o_i$$

$$\mu^2 = \bigcup_{i=1}^{N} \mu^2_i$$

A necessary and sufficient condition for a function generated by a PCLU is:

$$A \cdot M < T_1, \qquad M \in \mu^o \qquad (1)$$

$$A \cdot M \geq T_2, \qquad M \in \mu^2 \qquad (2)$$

The proof consists of developing an expression for the largest $A \cdot M$, $M \in \mu^o$ over all functions generated by Algorithm 6.2.2 obeying (d). It will be used to construct relations between T_1, T_2, and Δ such that the satisfaction of relations (1) and (2) is insured.

Example

Let $N = 5$, $J = 2$, and $X_2 = 00001$. Then the function has $\mathcal{S}_2^2 = \mu_2^2 = \{00011, 00101, 01001, 10001\}$.

After training for the first vector in μ_J^2, the weight component(s) a_x corresponding to the nonzero components of X_J will have a value $C\Delta$, C an integer, where

$$J\ C\Delta \geq T_2. \tag{3}$$

C represents the number of training trials required to drive the discriminant above T_2.

Example

After training is complete for 00011, the mf weight vector A will be $A = (0, 0, 0, 1, 1)C\Delta$.

Training for the second vector in μ_J^2 benefits from the previous training, since the discriminant at the start of training will already have a value $(J-1)C\Delta$. Therefore, the increment required of the appropriate weight components is $1/J(C\Delta)$. Of course C must contain J as a factor.

Example

After training is complete for 00101, $A = (0, 0, \frac{1}{2}, 1, \frac{3}{2})C\Delta$.

At the completion of all training, the components a_x have a magnitude:

$$a_x = C\Delta \sum_{i=1}^{N-J+1} J^{1-i}. \tag{4}$$

Example

After training is complete, $A = (\frac{1}{8}, \frac{1}{4}. \frac{1}{2}, 1, \frac{15}{8})C\Delta$.

In order to add Δ an integral number of times to a weight component, it is necessary that

$$C = J^{N-J}. \tag{5}$$

Furthermore, in order for the training to be successful, it is necessary that both X_J and the input of highest Hamming weight assigned a zero output, which will always be $1-X_J$, produce discriminants less than T_1. But since the weight components a_x are incremented every time any weight component is incremented, and the number of components a_x is at least equal to the number of other components incremented during any one training session, the discriminant for X_J will be at least as large as the discriminant for $1-X_J$. Therefore, it is necessary that

$$(J-1)\, a_x < T_1 \tag{6}$$

or,

$$(J-1)\, C\Delta \sum_{i=1}^{N-J+1} J^{1-i} < T_1. \tag{7}$$

<u>Example</u>

With $C = 2^{5-2} = 8$, $A = (1, 2, 4, 8, 15)\Delta$.

Note that $A \cdot (1 - X_J) = A \cdot X_J = 15\Delta$.

Relations (3), (5), and (7) are enough to insure that training will be successful if the functions are of the kind discussed so far. They can be used to compute the values to be assigned to Δ, T_1, and T_2 of the PCLU before training begins. QED

<u>Example</u>

$$16\Delta \geq T_2. \tag{3}$$

$$15\Delta < T_1. \tag{7}$$

$T_2 = 10^4$, $T_1 = 9.8 \times 10^3$, and $\Delta = 650$ satisfy these inequalities.

If an Algorithm 6. 2. 2 function does not obey (d), then the expression for the largest $A \cdot M$, $M \epsilon \mu^{o}$ can be awarded to either $A \cdot X_J$ or $A \cdot (1-X_J)$, as the following examples demonstrate.

<u>Example 6. 2. 1</u>

Let

$$\mathcal{S}_2^2 = \{(000011)\}$$

$$\mathcal{S}_3^2 = \phi$$

$$\mathcal{S}_4^2 = \phi$$

$$\mathcal{S}_5^2 = \{(111101)\}$$

The mf weight vector after training is: $A = (\frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, 1, \frac{6}{5})C\Delta$.

Therefore, since $X_2 = (000001)$,

$$A \cdot (1-X_2) = 9/5\ C\Delta > A \cdot X_2 = 6/5\ C\Delta.$$

Example 6. 2. 2

Let

$$\mathcal{S}_3^2 = \{(0000111),\ (0001011)\}$$

$$\mathcal{S}_4^2 = \{(0110011),\ (1010011)\}$$

The mf weight vector after training is:

$$A = (1,\ 4,\ 5,\ 16,\ 48,\ 69,\ 69)C/48\,\Delta .$$

Therefore, since $X_3 = (0000011)$,

$$A \cdot (1-X_3) = 74/48\ C\Delta < A \cdot X_3 = 138/48\ C\Delta .$$

Therefore, for the general Algorithm 6. 2. 2 function,

$$\max\ \{ (J-1)\, a_x,\quad A \cdot (1-X_J) \} < T_1 .$$

The following property includes the specific values for this expression.

Property 6. 2. 3

For every PCLU in the net, if

(a) Conditions 6. 2. 2 are satisfied,

(b) $J\, C\Delta \geq T_2$, J as defined in Property 6. 2. 2,

(c) $\max\{(J-1)\,a_x,\ A \cdot (1-X_J)\} < T_1$, where

$$a_x = C\Delta \left[\sum_{i=1}^{|\mathcal{S}_J^2|} J^{1-i} + J^{1-|\mathcal{S}_J^2|} \times \sum_{\substack{\ell \\ \text{see} \\ \text{C1}}} \left(\prod_{\substack{j \\ \text{see} \\ \text{C2}}} j^{-|\mathcal{S}_j^2|} \sum_{i=1}^{|\mathcal{S}_\ell^2|} \ell^{-} \right) \right]$$

$$A \cdot (1-X_J) = C\Delta \left[\sum_{i=1}^{|\mathcal{S}_J^2|} J^{1-i} + J^{1-|\mathcal{S}_J^2|} \times \sum_{\substack{\ell,k \\ \text{see} \\ \text{C3}}} (\ell-k+1) \left(\prod_{\substack{j \\ \text{see} \\ \text{C2}}} j^{-|\mathcal{S}_j^2|} \sum_{i=1}^{|\mathcal{S}_\ell^2|} \ell^{-i} \right) \right]$$

C1 -- This sum is over all subscripts ℓ of $\mathcal{S}_\ell^2$ where $\ell \geq J+1$ and $\mathcal{S}_\ell^2 \neq \phi$.

C2 -- This product is over all j, $J < j < k$ such that $\mathcal{S}_j^2 \neq \phi$.

C3 -- In addition to the ℓ defined in C1, k is the largest k' for which $^2_{k'} \neq \phi$ and yet $k' < \ell$.

(d) $C = J^{|\mathcal{S}^2_J| - 1} \prod_K K^{|\mathcal{S}^2_K|}$

where the product is over all subscripts of $\mathcal{S}^2_K$ greater than J.

Proof:

At the completion of training for $\mathcal{S}^2_J$:

$$a_x(\mathcal{S}^2_J) = C\Delta \sum_{i=1}^{|\mathcal{S}^2_J|} J^{1-i}$$

$$A \cdot (1 - X_J) = D(\mathcal{S}^2_J) = a_x(\mathcal{S}^2_J).$$

If $|\mathcal{S}^2_J|$ had been one greater, say due to some input Y, then the amount of increase required in the discriminant $A \cdot X_J$ would have been

$$J \; J^{-|\mathcal{S}^2_J|} \; C\Delta .$$

This quantity would have been divided among the J - 1 a_x weight components and one other weight component whose value had remained zero up to that time.

The next input requiring training is in $\mathcal{S}^2_K$, $K > J$. It differs from Y only in having more than one other weight component which has remained zero. Therefore, the increase required to attain $J\,C\Delta$ is divided among K components, and the a_x increment is:

$$C\Delta\,(J/K)\; J^{-|\mathcal{S}^2_J|} . \tag{1}$$

If there is another element in $\mathcal{S}^2_K$, then it will differ from the preceeding vector in only two components in the same way two vectors are different in $\mathcal{S}^2_J$. Therefore, the discriminant of the new input before training is short of $J\,C\Delta$ by (1). If this value is divided evenly among the K weight components associated with nonzero input components, then the increment to any one weight component is:

$$C\Delta\,(J/K^2)\; J^{-|\mathcal{S}^2_J|} . \tag{2}$$

In general, after the completion of training for $\mathcal{S}^2_K$:

$$a_x(\mathcal{S}_K^2) = a_x(\mathcal{S}_J^2) + C\Delta J \cdot J^{-|\mathcal{S}_J^2|} \sum_{i=1}^{|\mathcal{S}_K^2|} K^{-i}$$

Each of the weight components selected by $X_K - X_J$ (there are K-J of these) are increased by the same amount as the a_x the sum of all other increments to all the remaining weights is equal to the a_x increment. Therefore

$$D(\mathcal{S}_K^2) = D(\mathcal{S}_J^2) + (K-J+1) \left[J \cdot J^{-|\mathcal{S}_J^2|} \sum_{i=1}^{|\mathcal{S}_K^2|} K^{-i} \right] C\Delta .$$

If another $\mathcal{S}_L^2$, $L > K$, is not empty, then, by the same reasoning as for the $\mathcal{S}_K^2$ case, the necessary total increment for the first input of this set must equal:

$$K \left[J \cdot K^{-|\mathcal{S}_K^2| -1} \quad J^{-|\mathcal{S}_J^2|} \right] \quad C\Delta .$$

This quantity is divided among L components. Therefore, the a_x increment is 1/L of this. In general, after training is complete for this set:

$$a_x(\mathcal{S}_L^2) = a_x(\mathcal{S}_K^2) + C\Delta K \left[K^{-|\mathcal{S}_K^2| -1} \quad J^{1-|\mathcal{S}_J^2|} \right] \times \sum_{i=1}^{|\mathcal{S}_L^2|} L^{-i}$$

and

$$D(\mathcal{S}_L^2) = D(\mathcal{S}_K^2) + (L-K+1)\, K^{-|\mathcal{S}_K^2|} \quad J^{1-|\mathcal{S}_J^2|} \times \sum_{i=1}^{|\mathcal{S}_L^2|} L^{-i} .$$

By an extension of this argument, the expressions at the completion of all training are those given in the statement of the property.

The property is proved if a technicality involving the integer C is cleared up. The smallest quantity a weight component can be increased by is Δ. In order for C to contain all factors that might occur during a training session and thereby allow an increase of Δ and no more, C should contain all of the factors given in (d). QED.

Example

The computation of (c) for Example 6. 2. 1.

$$a_x(C\Delta(2^0 + 2^0 \text{(no product term) } 5^{-1}) = (1 + 1/5)C\Delta.$$

$$a_x = 6/5\ C\Delta.$$

Therefore,

$$(J-1)a_x = (2-1)a_x = 6/5\ C\Delta.$$

$$A \cdot (1-X_j) = C\Delta(1 + 2^0(5-2+1)(1/5)) = 9/5\ C\Delta.$$

Therefore,

$$\max\{(J-1)\,a_x,\ A \cdot (1-X_J)\} = 9/5\ C\Delta.$$

Example

The computation of (c) for Example 6. 2. 2.

$$a_x = C\Delta\left(\sum_{i=1}^{2} 3^{1-i} + 3^{1-2}\text{(no product term) }\sum_{i=1}^{2} 4^{-i}\right).$$

$$a_x = 69/48\ C\Delta.$$

Therefore,

$$(J-1)\,a_x = 2a_x = 138/48\ C\Delta.$$

$$A \cdot (1-X_J) = C\Delta\ \{1 + 1/3\ (4-3+1)\ (1/4 + 1/16)\}$$
$$= 74/48\ C\Delta.$$

Therefore,

$$\max\{(J-1)a_x,\ A \cdot (1-X_J)\} = 138/48\ C\Delta.$$

CHAPTER 7
DISCUSSION

7.1. Summary

An automaton model of the CA3 sector of mammalian hippocampus is presented. The connectivity between the PCLU (the pyramidal cell model) rank and the BCLU (the basket cell model) rank is left unspecified except that a direct PCLU-BCLU-PCLU loop is required for each PCLU. It is assumed that whenever the output of a PCLU's delay is nonzero, the output of its special BCLU is also nonzero. The input to each PCLU is a vector M_i with components having values from the set $\{0, 1\}$. The output of the model is a time-sequence of vectors of the form

$$\sigma(M_i) = 0\, H^1\, H^1\, H^2\, H^2\, H^3\, H^3 \ldots,$$

with each vector H^j having components h_k^j with values from the set $\{0, 1, 2\}$. Assuming each nontrivial input is separated by a zero input to clear circulating quantities left over from the previous input, the output sequences are shown to have these properties:

1. Each sequence terminates in either an equilibrium or a cycle.
2. $h_k^1 \geq h_k^2, h_k^3, \ldots$
   $h_k^2 \leq h_k^3, h_k^4, \ldots$
   $h_k^3 \geq h_k^4, h_k^5, \ldots$, etc.
3. If $H^i = H^j$, $j > i$, then $H^i\, H^i \ldots H^{j-1}\, H^{j-1}$ is a cycle.

An algorithm is developed to generate all possible output sequences of any model containing N PCLUs.

The characteristics of a training structure for reshaping the output sequences of the foregoing model are also presented. This structure is supported by a target table containing a set of allowed output sequences for each input to the model. It is assumed that when the system is placed in its environment for the first time, the function realized by the model is contained in the target table. In order to insure this, a special training session is held before the model is placed in its environment. An algorithm (Algorithm 6. 2. 2) is developed to generate the function placed in the target table for the special session. If certain parameters (the mossy fiber and feedback weights) are set correctly (to zero) at the beginning of this session, the function realized by the model at the completion of training is the function in the target table.

After the system is placed in its environment, desired changes in the model's function are registered by changing the target table. The trainer compares the output sequence generated in response to a net input, M, with the sequences in the target table. If no match can be found (which implies a change in the target table has been detected), a marker is set. The next time M occurs as the net input, the output sequence up to the point of the fault is generated, and then a training session is triggered. It is proved that the training session is guaranteed to "succeed" if and only if both the change in the target table and the selection of some of the model's parameters (Δ, δ, T_1, and T_2) are in accordance with certain rules (Property 4. 4. 1). To "succeed," the output sequence must be the same as the target table sequence only up to and including the element containing the change. It is understood that the outputs following the subsequence just described, as well as any other output sequence of the model, may be altered by this training session. The model's new function may or may not be the same as the functions in the target table. If it is not the same, then, more training sessions are required.

A number of other ancillary results on the time-domain behavior of CA3-like automata were also obtained, both analytically and by computer simulation.

7. 2. Comments on the Neuroscientific Aspects of this Study

Assume that the hippocampus is a memory bank containing transformations of single inputs into output sequences, and that its task is to make act decisions. Furthermore, assume that a trainer is available for changing the output sequence that any input is transformed into and that it operates in the manner described in Chapter 5. The following observations might now be of speculatory interest to neuroscientists.

The results of Section 6.1 on training phases, together with Property 4.4.1, suggest an increase in both the capability of the trainer and the complexity of the hippocampus's transformations as it matures. At birth, and during phase 1, a single input is related to a single output; that is, the relevant output is not sequential. At this stage, the trainer can increase the firing rate of an output but not decrease it. As the hippocampus matures, and in particular as the basket cell rank begins to make connection with pyramidal cells, the outputs of the hippocampus can become sequential in nature, involving oscillations. The trainer now has the ability to decrease the output rate, but at the risk of forcing the output into oscillations. The trainer cannot yet suppress these oscillations. This capability is achieved only when the basket cells have made connections with a sufficient number of pyramidal cells.

A second observation is related to the assumed ability of the natural system to avoid training instabilities. Recall that in the model, successful training can be guaranteed if and only if certain rules are followed when altering the target table and certain relationships are obeyed when specifying the CA3 sector model's parameters. But even then undesirable changes can occur in other output sequences. In fact, it is possible that: (1) either these changes cannot be corrected; or (2) as each change is corrected, another mismatch occurs. Such training instabilities might be dangerous to an animal.

A third observation involves the problem the trainer has in selecting the output to be retrained when a mismatch occurs. As mentioned in Section 5.3, one approach would be to select the most "uncertain" PCLU. Another approach, involving training all PCLUs at once, might also be used.

A related observation involves the knowledge a hypothetical natural target table generator has of the connectivity of the natural functal net. From computer simulations, it appears that the more information the target table generator has about the connectivity, the more freedom it has in making changes in the target table that are guaranteed realizable by the net. On the other hand, the more connectivity knowledge the target table generator has, the greater the information that must be genetically stored and the greater the chance for a connectivity error to occur during growth. In the author's opinion, the weight of evidence supports only the most general kind of connectivity knowledge on the part of the natural target table generator, and hence supports a limited function changing capability with safety.

The final observation pertains to the code employed by the natural system to convey act information. If the hippocampus is indeed an act computer, there must be a direct relationship between behavior and the hippocampus's output. Since the behavior of a mammal often consists of essentially a stimulus-directed Markovian sequence of actions, each output of the hippocampus might well be related in a nontrivial way to its preceeding output. In other words, a hippocampus output associated with a certain behavioral act on one occasion may be associated with a different behavioral act on another occasion. The original function of the hippocampus would have to be compatible with this, as would the hypothetical target table generator when it decided on changes in the hippocampal output function.

7.3. Comments on the Engineering Aspects of the Study

The functual system theory developed in this report introduces a new perspective for understanding interconnected arrays of variable function nonlinear function generators (functals). Useful applications of this theory may arise in fields other than neurocybernetics.

It is generally accepted that the nervous system combines memory and logic in the same location in an extremely effective way. The Kilmer-McCulloch Retic model, the Kilmer-McLardy hypothesis of the task of the hippocampus, and the hippocampus model presented in this report suggest a partial organization of a robot controller which takes advantage of this property. Consider the design of the controller for a moon rover. The controller can be imagined as a hierarchy of subcontrollers with the apex occupied by the Retic, which commands the mode of the rover. As an example, suppose one of the modes is "proceed with the search."

The rover would receive information on its environment through its sensory transducers. A reasonable choice of transducers for a moon rover might be a 3-D television camera, temperature and pressure sensors (for internal state monitoring), and tactile sensors (on probes, shovels, and bumpers). The data from these would be fed into processors designed to extract certain kinds of information. Some of these may be assigned the task of processing data for input to the hippocampus system.

The hippocampus occupies the next level of the hierarchy; it computes the acts within a mode. For example, the acts within the "proceed with the search" mode might define the direction and speed of the rover and the search mode of its camera system. The acts associated with an input configuration would have to be programmed on earth according to the best information available. Once on the moon, however, if either a situation occurred which was found to be harmful to the rover or an unexpected situation occurred, then the hippocampus would be retrained. From the hippocampus the act command would be passed on to lower levels where the actual motor command sequences would be generated.

There are many problems yet to be solved while pursuing the details of any hippocampus system design for a robot. Most of these are analogous to problems yet to be solved in the natural system. Among these are:

(1) the definition of the code assigned to each output;
(2) a determination of whether the code is context-sensitive or context-free;
(3) the definition of an initial function for a net which affords the robot maximum protection and versatility;
(4) the specification of the connectivity of the net (Do usable connectivities exist which increase the freedom of the trainer?);
(5) the specification of the trainer rules to (a) guarantee successful training, (b) select the PCLU to be trained,

(c) select the direction in which the PCLU is changed, and (d) allow the new output to fit smoothly into the act sequence.

LIST OF REFERENCES

1. Von Bekesy, G., Sensory Inhibition (Princeton University Press, 1967).

2. Eccles, J. C., Ito, M., and Szentagothai, J., The Cerebellum as a Neuronal Machine (Springer-Verlag, New York, 1967).

3. Eccles, J. C., "Postsynaptic inhibition in the central nervous system," The Neurosciences (Gardner C. Quarton, Theodore Melnechuk, and Frances O. Schmitt, eds., The Rockefeller University Press, New York, 408-426, 1967).

4. Wilson, V. J., "Inhibition in the central nervous system," Scientific American 214, 102-110 (1966).

5. Scheibel, M. E. and Scheibel, A. B., "Spinal motorneurons, interneurons and Renshaw cells. A Golgi study," Arch. Ital. Biol. 104, 328-353 (1966).

6. Maturana, H. R., Lettvin, J. Y., McCulloch, W. S., and Pitts, W. H., "Anatomy and physiology of vision in the frog (Rana pipiens), J. Gen. Physiol. 43 (No. 6, Pt. 2), 129-175 (1960).

7. Ratliff, F., "On fields of inhibitory influence in a neural network," Neural Networks (E. R. Caianiello, ed., Springer-Verlag, New York, 6-23, 1968).

8. Barlow, R. B. Jr., and Levick, W. R., "The mechanism of directionally selective units in the rabbit's retina," J. Physiol. (Lond.) 178, 477-504 (1965).

9. Wilson, D. M., and Waldron, I., "Models for the generation of the motor output pattern in flying locusts," Proceedings of the IEEE 56, 1058-1064 (1968).

10. Ratliff, F., and Mueller, C. G., "Synthesis of "On-Off" and "Off" responses in a visual-neural system," Science 126, 840-841 (1957).

11. Hubel, D. H., and Wiesel, T. N., "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," J. Physiol. (Lond.) 160, 106-154 (1962).

12. Kilmer, W. L., "A circuit model of the hippocampus of the brain," AFOSR Scientific Report, Division of Engineering Research, Michigan State University (July 1970).

13. Kilmer, W. L., "The reticular formation: Part I, Modeling studies of the reticular formation; Part II, The biology of the reticular formation," AFOSR Scientific Report, Division of Engineering Research, Michigan State University (February 1969).

14. Kauffman, S. A., "Metabolic stability and epigenesis in randomly constructed genetic nets," J. Theoret. Biol. 22, 437-467 (1969).

15. Purpura, D. B., in Basic Mechanisms of the Epilepsies (Jasper, H. H., Ward, A. A., and Pope, A., eds., Little, Brown and Co., Boston, 1969).

APPENDIX A

BACKGROUND ON THE DEVELOPMENT OF THE HIPPOCAMPUS NET

A. 1. The Pyramidal Cell Logic Unit

The pyramidal cell model as originally conceived was the set of continuous firing rate equations shown in Figure A. 1. In this figure, Equation 2 says that the firing rate of model pyramidal cell j at the axon hillock at time t, $y_j(t)$ is a linear function of $x_j(t)$ only when $x_j(t)$ is in the range from 0 to α_{myj}. If $x_j(t)$ is less than zero, then $y_j(t) = 0$. If $x_j(t)$ is greater than α_{myj}, then $y_j(t)$ is equal to the maximum value of $\alpha_{myj} \cdot \alpha_{yj}$.

The function $x_j(.)$ as defined in Equation 1 consists of six terms:

1. $\sum_{i=1}^{I} \epsilon_{ji} \gamma_{ji} a_{ji}(t-\tau_{Aji}) z(t-\tau_{ji})$

This term represents the effect of the firing rates of the basket cells on the firing rate of the pyramidal cell. To explain the concepts which were used to develop this term, assume synaptic contact is made between basket cell i and pyramid j. At time $t-\tau_{ji}$ the basket cell fired at a rate $z_i(t-\tau_{ji})$. This signal traveled through various collaterals to bouton j, i, arriving there at time τ_{Aji}, altered by an amount γ_{ji}. (Note: By convention, if there is no connection between basket cell i and pyramid j, then $\gamma_{ji} = 0$.) At or near the bouton, the signal is modified by the memory process $a_{ji}(t-\tau_{Aji})$, which is defined in Equation 4. Finally, the signal passes through the dendritic arbor and soma of the pyramidal cell as an inhibitory post-synaptic potential and arrives at the axon hillock at time t, having been altered on the way by an amount ϵ_{ji}.

2. $\sum_{k=1}^{K} \sigma_{jk} s_k(t-\tau_{sjk})$

This term represents the effect of the septal fiber firing rate on the

$$X(t) = -\epsilon \gamma A(t-\tau_A) z(t-\tau) + \sigma s(t-\tau_s) + \theta M(t-\tau_M) + \beta Y(t-t_x) - \Gamma(t) \quad (1)$$

$$Y(t) = (y_1(t),\ y_2(t),\ \ldots\ y_J(t))^T \quad (2)$$

where

$$y_j(t) = \begin{cases} 0 & x_j(t) \leq 0 \\ a_{yj} x_j(t), & 0 < x_j(t) < a_{myj} \\ a_{myj} a_{yj}, & x_j(t) \geq a_{myj} \end{cases}$$

$$\Gamma(t) = \Psi \int_0^t \exp\left[-\frac{1}{\delta}(t-w)\right] x(w)dw + \Gamma_o \quad (3)$$

$$A(t) = 1 + (\Pi - 1)\exp\left\{-\frac{1}{T}\int_0^t \exp\left[-\frac{(t-w)}{\lambda}\right]\gamma z(w-\tau_z)dw\right\} \quad (4)$$

Figure A. 1. The pyramidal cell firing rate equations.

The expressions are for J pyramidal cells, I basket cells, K septal fibers, and N mossy fibers. The dimensions of the vectors are: $X(t)$, $\Gamma(t)$, Ψ, δ, Γ_o, a_{mx}, a_x, and $C : Jx1$; $Z(t) : Ix1$; $s(t) : Kx1$; $M(t) : Nx1$. The dimensions of the matrices are: $A(t)$, λ, Π, τ_A, τ, ϵ, $\gamma : JxI$; β, τ_x: JxJ; τ_M, $\theta : JxN$; τ, $\sigma : JxK$.

firing rate of the pyramidal cell. The memory effects between the septum and the cell are assumed to be constant relative to the basket to pyramidal cell memory. This will also be true of both the mossy fiber and other pyramidal cell inputs discussed below.

3. $\sum_{n=1}^{N} \theta_{jn} m_n(t-\tau_{Mjn})$

This term represents the effect of the mossy fiber input on the firing rate of the pyramidal cell.

4. $\sum_{\ell=1}^{J} \beta_{j\ell} y_\ell(t-\tau_{xj\ell})$

This term represents the input from other pyramids and the possible feedback from pyramid j itself.

5. $\Gamma_j(t)$

This term is the variable threshold defined by Equation 3. This expression is an attempt at a simple linear continuous equation for the kind of firing rate dependence on the input rate threshold above which nerve spikes are generated: the threshold increases as the firing rates of the inputs to the neuron increases in the recent past. The equation is a convolution of the potential function with an exponential decay. Thus, at some time t the threshold is made up of a constant term plus an infinite number of terms of the form

$$f(w) \exp\{-1/\tau\ (t-w)\} \qquad 0 \le w \le t.$$

Therefore, the value of the potential function which occurred at time $w = 0$ will have decayed the most, since it would have the value

$$f(0) \exp(-t/\tau);$$

and the value of the potential function occurring at time $w = t$ will have decayed not at all, since it would have the value

$$f(t) \cdot 1 = f(t).$$

Equation 4 is an attempt to give the pyramidal cell model a memory, where memory can be loosely defined as a device for storing records of events which have occurred in time previous to the present.

The a_{ji}-th entry expresses the concept that the memory process becomes larger as the basket cell i's firing rate $z_i(t-\tau_z)$ in the recent past becomes larger and approaches 1 in the limit: that is, as

$$A = \int_0^t \exp\left[-\frac{(t-w)}{\lambda_{ji}}\right] \gamma_{ji}\, z_i(w-\tau_{xji})dw \rightarrow \text{large},$$

$$\exp[-A/T_{ji}] \rightarrow 0 \quad \text{and} \quad a_{ji}(t) \rightarrow 1$$

If the basket cell i's firing rate $z_i(t-\tau_z)$ has been very small in the past, then $a_{ji}(t)$ approached some minimum value Π_{ji}: that is, as the A expression defined above becomes small,

$$\exp(-A/T_{ji}) \rightarrow 1 \quad \text{and} \quad a_{ji}(t) \rightarrow \Pi_{ji}.$$

The PCLU as defined in Chapter 3 is an extreme simplification of this continuous model. Some more of the more important simplifying assumptions are:

1. There is a constant threshold.
2. There are no inputs from other PCLUs.
3. The septal input controls the magnitude of A and is not of primary importance in the determination of the pyramidal cell firing rate.
4. The memory has no decay.
5. Most time lags are omitted.

A. 2. The Basket Cell Logic Unit

The terms in the basket cell firing rate equations, Figure A. 2, are analogous to terms in the pyramidal cell firing rate equations. Z(t), Equation 2, is the basket cell firing rate vector. It is expressed in the same form as Y(t), with α_{zi} being the proportionality constant and α_{mzi} being the maximum permissible value of $d_i(t)$.

D(t) is the basket cell firing rate potential vector, and it is analogous to X(t). The first term on the right hand side of Equation 1 is of the same form as Equation 1, Figure A. 1, term 1: ϕ corresponds to ϵ; Δ corresponds to γ; $G(\cdot)$ corresponds to $A(\cdot)$. The second term in the expression, $\Omega(t)$, is the threshold for the basket cells. Its Equation 3 is analogous to Equation 3 of Figure A. 1. The last term in the expression, ζ, is the constant firing rate potential vector for the basket cells, and it is analogous to C of the pyramidal cell expression. The memory expression, Equation 4, is of the same form as the memory expression for the pyramidal cells.

$$D(t) = \phi \, \Delta \, G(t-\tau_Q) \, Y(t-\tau_R) - \Omega(t) + \zeta \qquad (1)$$

$$Z(t) = (z_1(t), \ldots, z_I(t))^T, \qquad (2)$$

where

$$z_i(t) = \begin{cases} 0 & d_i(t) \le 0 \\ a_{zi} d_i(t), & 0 \le d_i(t) < a_{mzi} \\ a_{mzi} a_{zi}, & d_i(t) \ge a_m Z_i \end{cases}$$

$$\Omega(t) = \Omega_1 \int_0^t \exp\left[-\frac{1}{\rho}(t-w)\right] D(w) + \Omega_o \qquad (3)$$

$$G(t) = 1 + (\mu - 1) \exp\left\{-\frac{1}{\xi} \int_0^t \exp\left[-\frac{(t-w)}{\nu}\right] \Delta \, Y(t-\tau_V) dw\right\} \qquad (4)$$

Figure A. 2. The basket cell firing rate equations.

The expressions are for J pyramidal cells and I basket cells. The dimensions of the vectors are: $D(t)$, $Z(t)$, $\Omega(t)$, Ω_1, Ω_o, ζ, and ρ:Ix1; $Y(t)$: Jx1. The dimensions of the matrices are: ϕ, $G(t)$, τ_R, τ_Q, τ_V, ξ, μ, ν : IxJ; Δ : JxI.

It is clear from the BCLU model presented in Chapter 3 that some radical simplification of this model has been made. The major additional assumption for the BCLU over and above those presented in the previous section is that there is no memory process.

A. 3. The Connectivity

As originally conceived, the connectivity of the hippocampus model was based on the concept of a card. A card was defined as (1) all pyramidal cell (PC) models connected to one septal fiber, plus (2) all basket cell (BC) models which receive inputs from the PCs of the card (it was assumed that a BC did not receive inputs from two different cards), plus (3) a cell in CA1 which received inputs from every PC in the card. The output of the card was the output of this last cell. The communication between cards was accomplished by BC collaterals to the PCs of other cards.

This concept was modified to the connectivity described in Chapter 3, with one septal fiber per PC, because it seemed possible that a card could be modeled as a single PC.

APPENDIX B
A COMPUTER PROGRAM BASED ON ALGORITHM 4. 3. 1

Figure B. 1 is a CDC 6500 FORTRAN EXTENDED (MSU) listing of the program TTABLE and its subroutines. This program generates all possible output sequences of a CA3 sector net model containing N PCLUs. It does so according to Algorithm 4. 3. 1. Note that one data card is required in order to specify the number N; the format of this card is 10X, I5. The output sequences are printed in rows of ten; the format for a typical sequence is demonstrated by the following, which is an actual output sequence generated by TTABLE with N = 5:

```
PCLU 1-  1 0 1 0 1 0 1  (The output sequence component for PCLU 1.)
PCLU 2-  2 0 1 0 0 0 0
PCLU 3-  2 0 2 1 2 2 2
PCLU 4-  2 0 1 0 1 0 1
```

$$\text{PCLU 5-}\ \ 2\ 0\ 0\ 0\ \underbrace{0\ 0}_{\text{cycle}}\ 0.$$

Figure B.1: FORTRAN listing for TTABLE.

```
      PROGRAM TTABLE (INPUT, OUTPUT)
C........THIS PROGRAM COMPUTES THE OUTPUT SEQUENCE SET FOR A
C........HIPPOCAMPUS NET WITH N PCLUS.
C........IT DOES SO ACCORDING TO ALGORITHM 4.3.1.
C........VERSION 2
C           THIS VERSION COMPUTES THE SEQUENCES BEGINNING WITH ALL
C           TWOS AND WITH ALL TWOS EXCEPT ONE ONE.
      COMMON IOUT(10,10) , N ,IUP
C
C........PRELIMS.
      READ 200, N
      PRINT 201,N
      IUP = 50 / N
      NSQ = 1
      DO 5 I = 1, N
5     NSQ = NSQ * 3
C
C........GENERATE THE FIRST ELEMENT OF A TRIAL SEQUENCE.
      NA = NSQ - 1
      DO 10 I = NA,NSQ
      CALL GENOUT (I,1)
C........GENERATE THE SECOND ELEMENT OF A TRIAL SEQUENCE.
      DO 20 IA = 1,NSQ
      CALL GENOUT (IA, 2)
      CALL PROP202 (IOK, 2)
      IF (IOK.EQ.1) GO TO 20
      CALL PROP254 (IOK, 2)
      IF (IOK.EQ.1) GO TO 20
C........IF THE 2ND ELEMENT IS OK GENERATE 3RD.
      DO 30 IB = 1,NSQ
      CALL GENOUT (IB, 3)
      CALL PROP202 (IOK, 3)
      IF (IOK.EQ.1) GO TO 30
      CALL PROP254 (IOK, 3)
      IF (IOK.EQ.1) GO TO 30
C........IF THE 3RD ELEMENT IS OK GENERATE 4TH.
      DO 40 IC = 1, NSQ
      CALL GENOUT (IC, 4)
      CALL PROP202(IOK, 4)
      IF (IOK.EQ.1) GO TO 40
      CALL PROP254 (IOK, 4)
      IF (IOK.EQ.1) GO TO 40
C........IF THE 4TH ELEMENT IS OK GENERATE 5TH.
      DO 50 ID = 1, NSQ
      CALL GENOUT (ID,5)
      CALL PROP202 (IOK,5)
      IF (IOK.EQ.1) GO TO 50
      CALL PROP254 (IOK, 5)
      IF (IOK.EQ.1) GO TO 50
C........IF THE 5TH ELEMENT IS OK GENERATE 6TH.
      DO 60 IE = 1, NSQ
      CALL GENOUT (IE, 6)
      CALL PROP202 (IOK, 6)
      IF (IOK.EQ.1) GO TO 60
      CALL PROP254 (IOK, 6)
      IF (IOK.EQ.1) GO TO 60
C........IF THE 6TH ELEMENT IS OK GENERATE 7TH.
      DO 70 IF = 1, NSQ
      CALL GENOUT (IF, 7)
      CALL PROP202 (IOK, 7)
      IF (IOK.EQ.1) GO TO 70
      CALL PROP254 (IOK, 7)
      IF (IOK.EQ.1) GO TO 70
C........IF THE 7TH ELEMENT IS OK GENERATE THE 8TH.
      DO 80 IG = 1, NSQ
      CALL GENOUT (IG, 8)
      CALL PROP202 (IOK, 8)
      IF (IOK.EQ.1) GO TO 80
```

Figure B.1: (cont'd)

```
      CALL PROP254 (IOK, 8)
      IF (IOK.EQ.1) GO TO 80
C........IF THE 8TH ELEMENT IS OK GENERATE 9TH.
      DO 90 IH = 1, NSQ
      CALL GENOUT (IH, 9)
      CALL PROP202 (IOK, 9)
      IF (IOK.EQ.1) GO TO 90
      CALL PROP254 (IOK, 9)
      IF (IOK.EQ.1) GO TO 90
C........IF THE 9TH ELEMENT IS OK GENERATE 10TH.
      DO 100 II = 1, NSQ
      CALL GENOUT (II, 10)
      CALL PROP202 (IOK, 10)
      IF (IOK.EQ.1) GO TO 100
      CALL PROP254 (IOK, 10)
      IF (IOK.EQ.1) GO TO 100
C........IF THE 10TH ELEMENT IS OK PRINT THE ENTIRE SEQUENCE AND GO
C........NO FURTHER
      CALL BUFOUT (10)
100   CONTINUE
90    CONTINUE
80    CONTINUE
70    CONTINUE
60    CONTINUE
50    CONTINUE
40    CONTINUE
30    CONTINUE
20    CONTINUE
10    CONTINUE
C
200   FORMAT(10X,I5)
201   FORMAT(1H1,20X,*THE OUTPUT SEQUENCE SET FOR A HIPPOCAMPUS SET WITH
     1 *,I2,* PULSES*//)
      END
      SUBROUTINE PROP202(IOK,M)
C........THIS ROUTINE APPLIES PROPERTY 4.2.2 TO THE GENERATED SEQUENCE.
C
      COMMON IOUT(10,10) , N ,IOP
C
C........CHECK TO SEE IF M IS AN EVEN OR AN ODD NUMBER.
      IF((M/2).EQ.((M+1)/2)) GO TO 10
C........HERE FOR AN ODD NUMBER
      NU = M - 2  $NL = M - 1
      GO TO 11
C........HERE FOR AN EVEN NUMBER
10    NU = M - 1  $NL = M - 2
11    DO 20 I=1,N
      IF(IOUT(I,NU).LT.IOUT(I,M)) GO TO 12
      IF(M.EQ.2) GO TO 20
      IF(IOUT(I,M).LT.IOUT(I,NL)) GO TO 12
20    CONTINUE
C........HERE IF THE NEW ELEMENT IS OK.
      IOK = 0  $RETURN
C........HERE IF THE NEW ELEMENT IS NOT OK.
12    IOK = 1
      END
```

Figure B.1: (cont'd)

```
      SUBROUTINE PROP254(IOK,M)
C
C........THIS ROUTINE APPLIES PROPERTY 4.3.2 TO THE GENERATED SEQUENCE.
      COMMON IOUT(10,10) , N ,IOP
C
      IA = M - 1
      DO 15 J=1,IA
      DO 10 I=1,N
      IF(IOUT(I,M) .NE. IOUT(I,J)) GO TO 15
10    CONTINUE
C........HERE IF CURRENT ELEMENT EQUALS ANY PREVIOUS ELEMENT.
      IOK = 1
      CALL BUFOUT(M)   $RETURN
15    CONTINUE
C........HERE IF THE CURRENT ELEMENT DOES NOT EQUAL ANY PREVIOUS ELEMENT.
      IOK = 0
      END

      SUBROUTINE GENOUT (I,J)
C
      COMMON IOUT(10,10) , N ,IOP
C........THIS ROUTINE GENERATE A BASE 3 NUMBER FROM THE NUMBER I - 1.
C
      KC = I - 1
      DO 20 L = 1, N
      KA = KC/3
      KB = KC - KA * 3
      IOUT (L, J) = KB
      IF (KA) 10, 11
10    KC = KA
20    CONTINUE
40    CONTINUE
      RETURN
```

```
11     IF (L.EQ.N) GO TO 35
       L = L + 1
       DO 30 LL = L, N
       IOUT (LL, J) = 0
30     CONTINUE
35     CONTINUE
       GO TO 40
       END

       SUBROUTINE BUFOUT(M)
C........THIS SUBROUTINE IS THE OUTPUT ROUTINE FOR TTABLE.
       COMMON IOUT(10,10) , N ,IOP
       DIMENSION ISTOR(10,10)
       DATA NUM/1/,IPRNT/1/
C
       DO 10 I=1,M
C         PLACE THE M ELEMENTS OF THE I-TH PCLU OUTPUT SEQUENCE IN ONE
C         WORD OF ISTOR. IF M .LT. 10 , REMAINING BYTES BLANKED.
       ENCODE(M , 100 , ISTOR(I,NUM) ) (IOUT(I,J),J=1,M)
10     CONTINUE
       IF(NUM.GE.10) GO TO 11
       NUM = NUM + 1   $RETURN
11     NUM = 1
       IF(IPRNT - IOP) 12,12,13
13     PRINT 101,N
       IPRNT = 0
12     IPRNT = IPRNT + 1
       PRINT 102
       DO 20 I=1,N
       PRINT 200,I,(ISTOR(I,J),J=1,10)
20     CONTINUE
C
200    FORMAT(5X,*PCLU *,I2,10(2X,A10))
100    FORMAT(10I1)
101    FORMAT(1H1//20X,*OUTPUT SEQUENCES- *,I2,* PCLUS *)
102    FORMAT(1X//)
       END
```