THE DYNAMICS OF LEARNING IN SOME DIGITAL NETWORKS

by

MICHAEL CHRISTOPHER FAIRHURST

A thesis submitted for the degree
of Doctor of Philosophy at the
University of Kent at Canterbury

1 9 7 3

ABSTRACT

This thesis is concerned with a study of learning in feedback networks of adaptive logic circuits. Random networks have been studied by various researchers, but previous work has not considered the adaptation mechanisms in dynamic logic networks which result from exposure to a non-random environment.

Starting with a consideration of some limitations of a single-layer static network, the concept of a dynamic net (i.e. one with feedback connections) is introduced. The behaviour of the system is described in terms of its cycling activity in state space, and the effect of training on the state structure is considered.

Subsequent experimental investigations consider unsupervised learning in the net where early evidence of a clustering effect is seen. This effect is found to be more pronounced when constraints are applied to the system in the sense that controlling gates are included in the feedback path. The nature and definition of memory and perception in such nets, and the response of the net to sequences of inputs is also presented and discussed. In conclusion, a simple probabilistic analysis is developed so as to provide a basis for a general understanding of dynamic networks of this kind.

ACKNOWLEDGEMENTS

I would like to express my sincere thanks to Dr. I. Aleksander for his invaluable help and encouragement, and the interest which he has shown in this project. I would also like to acknowledge the support of the Science Research Council over the past three years.

Thanks are due to the Staff of the Electronics Laboratories at the University of Kent for providing research facilities and for helpful advice on many occasions, and in particular I am indebted to my colleagues in the Computer and Control research group.

Finally, I would like to thank Miss D. Paine for her careful typing of the text and printing of this thesis.

M. C. FAIRHURST

Canterbury

January 1973

CONTENTS

Page

CHAPTER 5

RECOGNITION OF PATTERN SEQUENCES.

CHAPTER 6

ANALYSIS AND DISCUSSION.

Page

CHAPTER 1

ARTIFICIAL INTELLIGENCE AND BRAIN MODELLING

1.1 Introduction.

The ability to learn (to modify behaviour in the light of past experience) is perhaps the most fundamental characteristic which one seeks as evidence of "intelligent" behaviour, whether in living or artificial systems. It is necessary, therefore, to see the work to be described here, which is concerned with the electronic simulation of learning, against the background of so-called "artificial intelligence" and brain modelling.

There seems to be some evidence that discussion and speculation on "artificial intelligence" and "thinking machines" dates back as far as the time of Babbage (see Bowden, 1953)[(1)], but it was only with the coming of large and fast digital computers that the implications of these speculations were first seriously considered.

The question "Can machines think?" held a fascination for researchers long before practical developments in artificial intelligence became feasible. The most notable contribution to the early literature on the subject came from Turing (1950)[(2)], who avoided the conventional approach of defining the terms "thinking" and "machine" and applied instead what has come to be known as the Turing test. Essentially, this proposes that if an external observer, communicating by some remote means (e.g. via a teletype), is unable to distinguish between a human subject and a machine, then the machine can be considered intelligent. In a more recent discussion, Armer (1960)[3] suggests that both machines and humans have a place in a multi-dimensional "continuum" of intelligence. In terms of sophistication, for instance, humans are clearly superior to machines, while the reverse is the case when considering relative positions in, say, the speed dimension.

The term "artificial intelligence" now embraces such a wide range of overlapping interests that the task of defining particular categories is almost impossible. One important area of research is that of heuristic programming. Early work in this field included the chess-playing program of Newell, Shaw and Simon (1958)[4], and Samuel's illustration of artificial learning using the game of checkers (Samuel, 1959)[5]. More recent work has produced programs for such diverse tasks as balancing assembly lines (Tonge, 1965)[6] and analysing chemical structures (Buchanan et. al., 1969)[7]. A comprehensive survey of heuristic programming techniques and achievements can be found in the book by Slagle (1971)[8].

A rather less fruitful approach to artificial intelligence is that of artificial evolution, the main proponents of this idea being Fogel, Owens and Walsh (1966)[9]. Rather than modelling <u>observed</u> phenomena occurring in nature, they attempt to simulate evolutionary processes which might occur. The modelling takes the form of producing generations of automata by means of selection and mutation. This artificial speeding up of evolution was found to be considerably hampered by the increasing complexity of successive "offspring".

Another important line of research comes under the general heading of intelligent networks, and involves studies of the bulk properties of groups of inter-connected but distinct individual elements. It is this approach which is pursued in subsequent pages, and hence a more detailed survey follows in section 1.2.

For a further study of the general field of artificial intelligence, several review papers can be consulted. Early work (mainly heuristic techniques) is surveyed by Minsky (1961)(10), who attempts to divide the field into its main sub-categories, and Solomonoff (1963)(11), while a later paper by Feigenbaum (1969)(12) reviews more recent work and lists some of the major centres of artificial intelligence research. A review of self-organising systems (usually networks of real or artificial neurons) is given by Pask (1964)(13), and an introduction to the formal theory of learning machines can be found in the book by Nilsson (1965)(14). For a closer study of pattern recognition, a research area with which the present work is connected, reference may be made to the book by Bongard (1970)(15) and a collection of papers on pattern recognition methodology edited by Watanabe (1969)[16].

1.2 Introduction to "intelligent" networks.

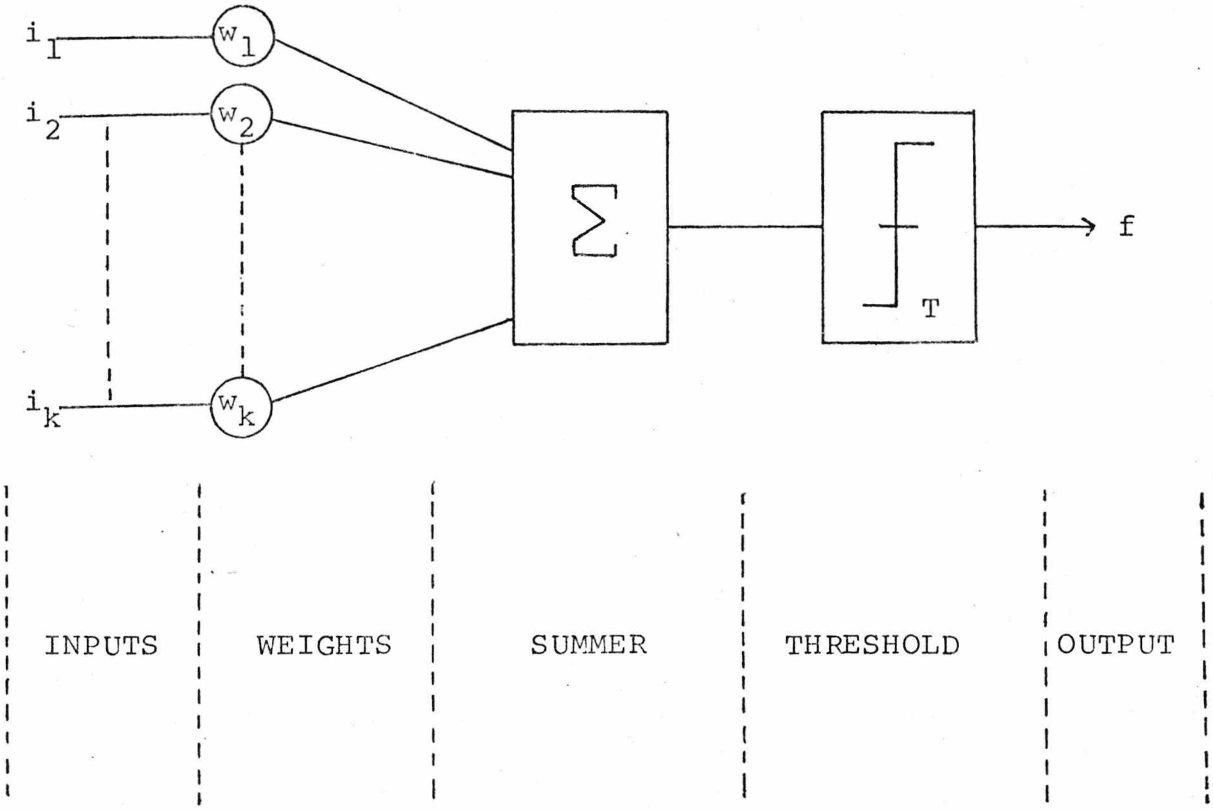
For many years the problem of designing artificially intelligent machines has prompted researchers to study some aspects of human intelligence as being likely to improve their designs. Within the spectrum of these endeavours to understand human methods of problem solving, one or two general references will serve to indicate the broad approaches. Travis (1963)[17] suggests introspection as a useful tool in such research, while typical psychological methods are illustrated in a collection of papers edited by Wason and Johnson-Laird (1968)[18]. The interaction between natural and "artificial" intelligence is illustrated (with particular reference to the pattern recognition problem) in a book subtitled "Studies in living and automatic systems" (Kolers, 1968)[19].

In the forefront of the network approach, however, have been the efforts to produce brain models capable of reproducing some of the intelligent functions of the human brain. Comparisons between the brain and the electronic computer (von Neumann, 1958[20], George, 1961[21], Wooldridge, 1963[22]) are numerous, but largely irrelevant in present context. It is more fruitful to trace the growth of interest in modelling the <u>functions</u> the brain - i.e. the behaviour of neural nets.

In addition to the neuropsychological theories aimed at understanding collections of individual neurons (characterised by Hebb's (1949) theory of form perception[23] - postulating the formation of "cell assemblies" and later "phase sequences" as a result of repeated stimulation of specific receptors - and simulated by Farley and Clark (1954)[24] and Rochester et. al. (1956)[25]) came the advancement of interest in the mathematical modelling of neuronal activity.

Fundamental to this work was the simple neuron of McCulloch and Pitts (1943)[26] - see Figure 1.1 - which makes use of the essentially binary nature of real neurons (see e.g. Brazier, 1951)[27]. Using the McCulloch and Pitts neuron model as the basic elements, Rosenblatt (1962)[28] proposed networks of artificial neurons which he called "perceptrons", a concept later studied in considerable depth by Minsky and Papert (1969)[29]. The perceptron concept was also to be found in several practical developments, such as the ADALINE of Widrow and Hoff (1961)[30] and the pattern recognition machine designed by Taylor (1959)[31], both of which incorporated the variable (analogue) weight principle in order to achieve a learning mechanism.

The classic work on mathematical modelling of neural processes is that of Caianiello (1961, 1966)[32],[33]. His method involves two sets of equations - <u>neuronic equations</u> representing the instantaneous behaviour of the system with fixed element functions, and <u>mnemonic equations</u>



Condition for neuron to fire is given by:

$$f = 1 \quad \text{when} \quad \sum_{j=1}^{k} w_j \, i_j \geqslant T$$

Figure 1.1: *Simple neuron model of McCulloch & Pitts.*

which describe changes in the functions themselves. The solutions of the neuronic equations predict a cyclic activity which is taken to represent the "thought processes" occurring in the network. This cyclic behaviour has much in common with the dynamic behaviour of the networks studied in the work to be described here.

1.3 Adaptive logic circuits.

The networks which are to be studied in this and the following chapters are not composed of artificial neurons in the accepted sense. The basic unit is the s.l.a.m. (stored logic adaptive microcircuit) element. Although not strictly a neuron model, it will be shown in later chapters that networks of these elements can function in a way comparable to more conventional "neural" nets.

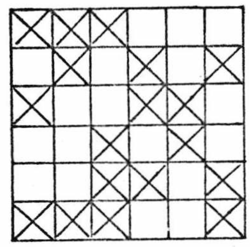
In operation, this element - which is a commercially available integrated circuit - is essentially identical to a bit-addressed (i.e. single output) random access memory (r.a.m.). There is already much published literature on the development, properties and applications of s.l.a.m. devices (see Aleksander, 1967[34], Aleksander and Albrow, 1968[35],[36]), but a functional description of their operation is given for reference in Appendix I.

The following section describes how in a simple experiment in pattern recognition, the learning behaviour of a group of human subjects can be modelled by a single-layer network of s.l.a.m. elements. The limitations of the comparison will provide the basis for the further experimental investigations reported in later chapters, which discuss much more flexible dynamic systems.

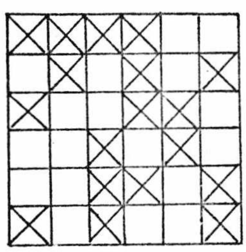
1.4 An experiment in human pattern learning.

One of the most common pattern recognition tasks which humans are required to perform in everyday life is that of reading and understanding letters and numerals, either printed or handwritten. Since this experiment is intended to assess some of the learning capabilities of humans, the use of such familiar characters in an experiment of this nature is precluded as they are already known to the subjects. However, just as everyday visual characters belong to a (very large) set of variants of "prototype" characters, it is possible to produce a random (and therefore unfamiliar) prototype pattern from which can be derived a set of "noisy" or distorted variants.

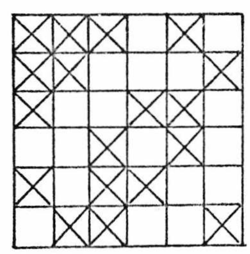
This prototype (the generating pattern) is shown in Figure 1.2(a), and from it were derived five groups of patterns, classified according to their Hamming distance (H) from the generating pattern. (Hamming distance as a measure of similarity between patterns is explained fully in section 2.3). The patterns were represented as



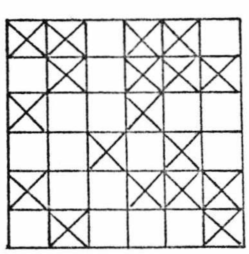
(a) Generator



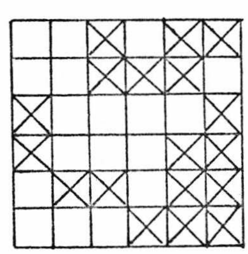
(b) H = 2



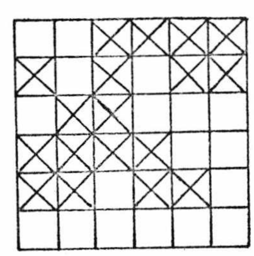
(c) H = 6



(d) H = 10



(e) H = 22



(f) H = 28

Figure 1.2: *Generator pattern and typical derivatives.*

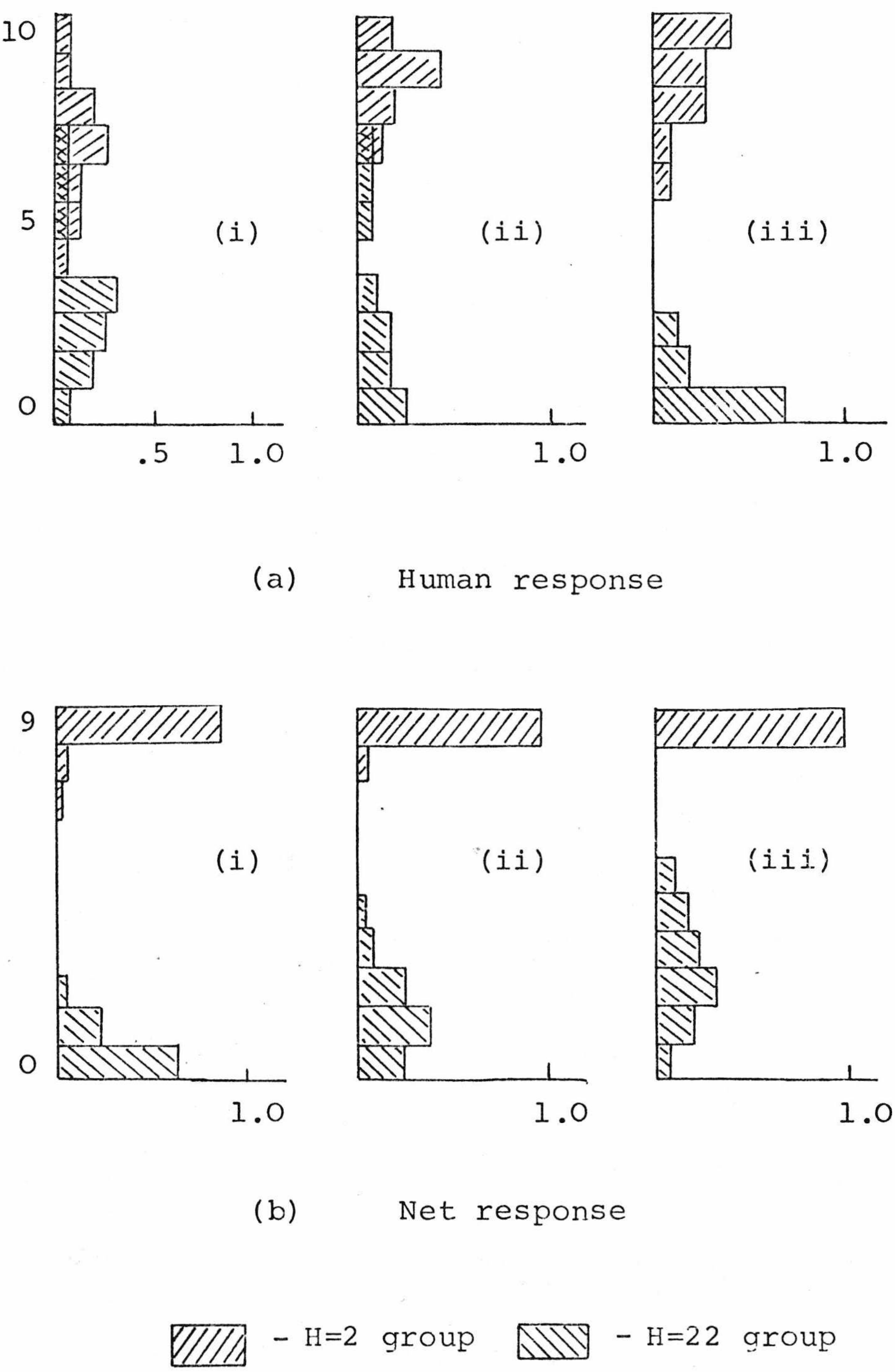
binary values in order that they could be used as inputs to a network of s.l.a.m. elements. For ease of viewing they were organised in a matrix as shown in Figure 1.2.

One group of patterns was chosen as the training set. These were each at a maximum Hamming distance of 2 from the generator and thus at a Hamming distance of 4 from each other. The other values of H used were 6, 10, 22 and 28. Representative examples from each of these five pattern groups are shown in Figure 1.2(b) to Figure 1.2(f).

The patterns were presented to a group of 16 volunteer subjects by projection on a screen. The experiment consisted of ten tests. In each test the first five patterns shown were training patterns (H=2), and the subjects were told that these patterns all belonged to the same class. These were followed by five test patterns, one from each of the five H-groups, and the subjects were asked to assess the similarity between each test pattern and the patterns of the training set. Their judgement was recorded by assigning to each test pattern a "score" between 0 and 10, a high score signifying a high degree of similarity between test pattern and training patterns, and a low score a low degree of similarity. Using this procedure the ten test runs were carried out, at the end of which the subjects had eventually been "trained" on fifty patterns.

In general it was found that high scores were given to similar patterns and low scores to dissimilar patterns, and that performance improved as a function of the amount of training. Some typical results, previously reported by Aleksander and Fairhurst (1970)[37], are repeated in Figure 1.3(a). This shows the response distributions as a function of training for test patterns of the two groups with Hamming distance values of 2 and 22 respectively, these representing patterns which are (in terms of Hamming distance) quite distinct.

If the Hamming distance relation between a test pattern and a set of training patterns is known, it is possible to predict the response of a single-layer network of s.l.a.m. elements (see Aleksander, 1970[38]). The distribution of responses for a net of nine 4-input elements was calculated (this giving the best fit to the observed experimental data), and is shown for comparison in Figure 1.3(b). It will be seen that while the net is able to distinguish clearly between the two pattern groups after training on only five patterns, the human response shows some confusion in the early stages (i.e. there is some measure of overlap between the distributions). The human response is seen to separate the two groups as more training patterns are seen, while in contrast, the electronic network will eventually saturate - i.e. produce a 100% response to <u>any</u> input pattern - if training is continued to excess.



(i) 5 training patterns

(ii) 20 training patterns

(iii) 35 training patterns

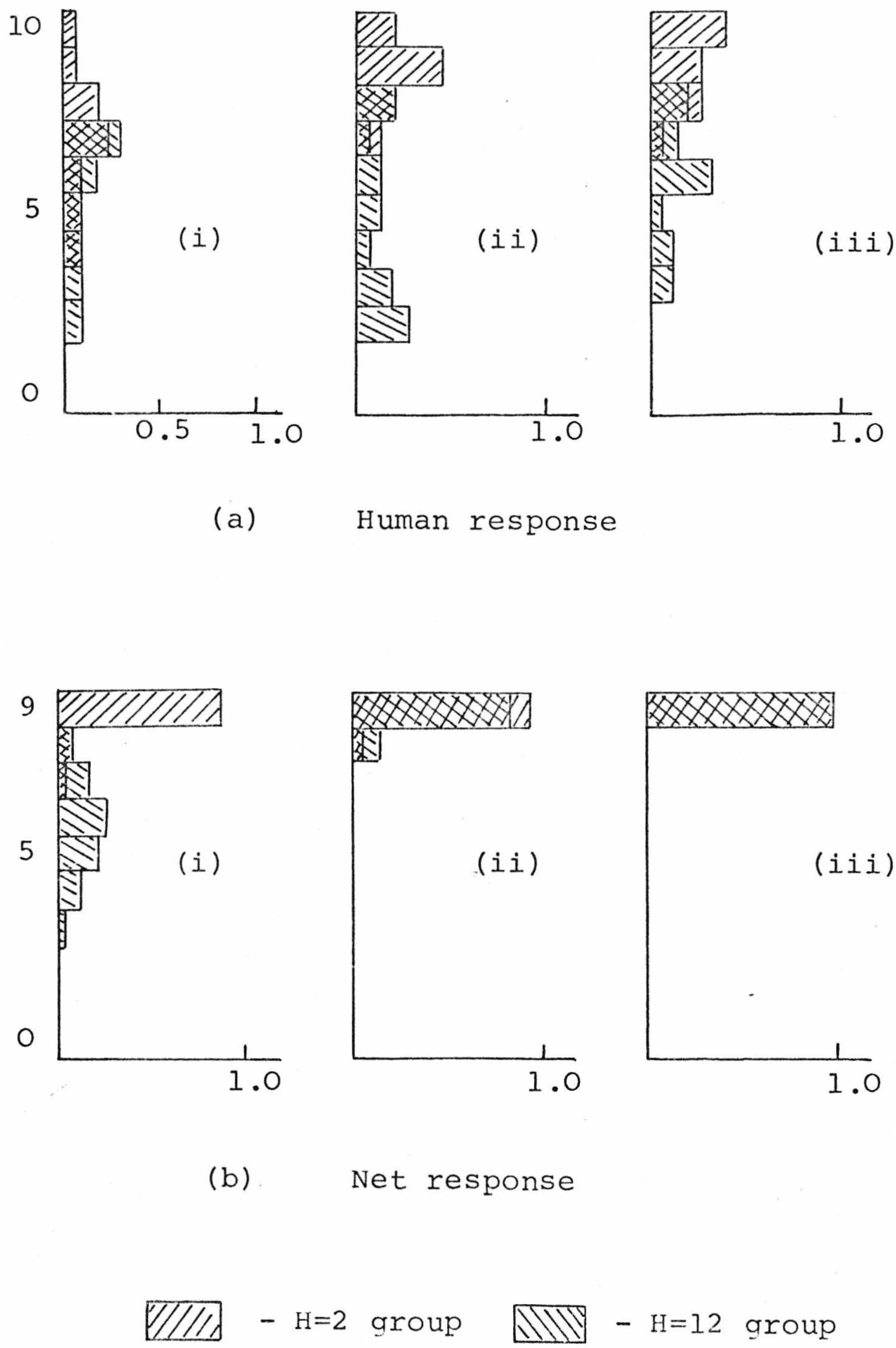
<u>*Figure 1.3:*</u> *Comparison of human response and net response.*

The results for other Hamming distance groups are similar in form and will therefore not be discussed at greater length. For completeness the distributions for two other groups - the H=2 and H=12 groups - are compared in Figure 1.4(a) and Figure 1.4(b). Here the net is completely unable to separate the two groups after training on 35 patterns, while human response shows a relatively high degree of distinguishability.

In the final part of the experiment the subjects, having seen fifty slightly degraded versions of a "prototype" (the generator pattern), were asked to draw their reconstruction of what they considered the generator pattern to be. Two attempts produced were completely correct and most of the remainder came within a Hamming distance of 10 of the correct pattern. Such behaviour is, of course, outside the scope of the simple electronic network considered earlier. The human capability of internally generating a "prototype" pattern can only be accounted for by assuming some kind of feed-back mechanism. Such a mechanism can easily be incorporated into the electronic system, and it is on these <u>dynamic</u> nets that attention will be focussed in later chapters.

1.5 Applications of automata theory to brain modelling.

There exist many theories and models (usually realised by computer simulation) which attempt to explain and



(i) 5 training patterns
(ii) 20 training patterns
(iii) 35 training patterns

Figure 1.4: Comparison of human and net response for different test groups.

imitate some aspects of the functioning of the brain. In particular, as is the case in the work to be reported here, much attention has been given to the modelling of the pattern recognition processes of the brain (e.g. visual perception - see Epstein, 1967[39]).

One of the most frequently pursued lines of attack is to consider the information processing system of the brain (visual, auditory etc.) as a collection of "black boxes", each labelled according to its specific function.

Typical of this approach is the model of the human visual networks proposed by Baron (1970)[40] which emphasises the distinctions between the operations of search, storage, recognition etc. (as compared with other visual processing models such as that of Kabrisky (1966)[41]). Noton (1970)[42] describes how patterns can be stored in memory as traces which contain both the pattern features and the shifts of attention from point to point in the visual field which relate them. Sparkes' model (1969)[43] for auditory pattern recognition, while leaving the precise mechanics of perception (e.g. detection of lines, angles, etc. - see Hubel and Wiesel, 1962[44] and Lettvin et. al., 1959[45]) unspecified, incorporates the idea of associative memory which is of interest in the context of what is to follow. For a survey of some formal models of visual and auditory perceptual systems and some related problems, the volume edited by Wathen-Dunn (1967)[46] should be consulted.

The research which is described in the following pages has its roots not in the "black box" approach, but in automata theory (Aleksander, 1970[47]). It is clearly possible to model the brain using standard automata theory concepts (see e.g. Booth, 1968[48] or Arbib, 1969[49]) and moreover, the model can be related to a <u>physical structure</u> which is realisable in electronic hardware (such as s.l.a.m. elements), providing a basis for explanation of several aspects of human behaviour such as adaptation, recognition and recall of environmental events etc. (Aleksander and Fairhurst, 1972[50]). When considered in this way the "thought processes" and memory mechanisms of the model can be conveniently expressed in terms of state transitions of the automaton. It is the precise nature of the possible state trajectories occurring under certain conditions which is the subject of the experimental investigations which follow. Some preliminary investigations in this field and a theoretical study of sequential adaptive logic circuits has previously been carried out by Mamdani (1971)[51].

The application of state space concepts to problems in artificial intelligence is not without precedent. For instance, Kiss (1967, 1969)[52],[53] uses the technique in his studies of models of word storage, as does Kauffman (1969)[54] when discussing the biological implications of random logic networks. It is worth considering the latter work in a little more detail since it provides a convenient background for much of the present work.

In his studies of networks of randomly interconnected elements, each with randomly assigned logic functions, Kauffman found that only few modes of activity (observable as repeating cycles of states) could exist. Typically, for a net of 100 two-input elements, the number of possible state cycles was found to be about 10 and the number of transitions per cycle to be of the same order. Kauffman comments on this remarkable "stability" in a biological context by considering the elements of the net as analogues of biological genes. The modes of behaviour (the state cycles) are then seen as the possible cell types arising from the interaction of the genes. Kauffman considers only random (untrained) networks in his experimental findings. The present research goes further by exploring the changes in behaviour (i.e. modifications to the state space structure) which occur when such a network is exposed to an environment which is <u>not</u> random.

The capacity to learn which is found in humans has produced many psychological theories (Hilgard, 1948[55], Borger and Seaborne, 1966[56]) since it is crucial to the study of intelligence. Similarly, the study of electronic learning and adaptation may prove crucial to an understanding of "artificial" intelligence in adaptive networks.

Fundamentally, the information processing operation of s.l.a.m. networks employs the n-tuple sampling method first introduced by Bledsoe and Browning (1959)[57], also investigated by Ullman and Kidd (1969)[58] and Ullman (1969)[59]. Here however, these techniques are extended by the introduction of feedback pathways within the system.

The guiding principle of the present work is that Stimulus-Response models (e.g. Hull, 1952[60])are to be discarded and replaced by networks which are dynamic in nature. The models of Ashby (1952, 1956)[61],[62] and Young (1969)[63] introduce the idea of homeostasis (self-regulation in biological systems) and the importance of feedback is also apparent in the information-flow models of MacKay (1956)[64] and Thomas (1970)[65].

Similarly, the concept of feedback in psychological research (where it is usually re-labelled "knowledge of results" - see Annett (1969)[66]) has been shown to be an important factor in human performance in many tasks (e.g. improvement in reaction times of subjects with knowledge of results, reported by Church and Camp (1965)[67]).

The following chapters will show how feedback processes within adaptive networks enable the traditional concepts of learning and adaptation to be redefined in a new light.

IMPORTANT NOTE:

One final word must be included here about experimental techniques, which applies to the remainder of the thesis. In general, results are quoted for one network only (i.e. one possible set of interconnections). It should be stated here that in all cases the results were confirmed by repeating experiments with a different connection matrix, but for clarity it has been considered unnecessary to include data which is redundant.

CHAPTER 2

CHARACTERISTICS OF AN UNSUPERVISED NETWORK

2.1 Learning without an external teacher

In most applications of adaptive - i.e. "learning" - networks (e.g. in problems of pattern recognition), it is assumed that an "intelligent" teacher is available (Nagy, 1968)[68]. The role of the teacher is to supply information about the desired response of the network to patterns in a training set, as a result of which the network modifies its behaviour so as to map input classes into a distinct set of outputs.

Nevertheless, it is possible for the learning process to be accomplished without external aid. (Spragins, 1966[69], Fralick, 1967[70]). This generally implies that the system is formulating information about the pattern classes from the statistical characteristics of the inputs presented to it.

One motivation for pursuing this idea is that in living systems "teach" signals must be extracted from the sensory information being received by the organism. Although much is known about the mechanisms of transmission of sensory data (see Lowenstein, 1966[71], Mellon, 1968[72]), there seems to be no evidence of separate pathways in the nervous system for "teach" information.

The idea of intermixing "teach" inputs and sensory inputs in the learning process in a simple single-layer s.l.a.m. net has been briefly introduced elsewhere (Aleksander, 1970)[73]. This chapter considers in greater depth the learning behaviour of a network which utilizes only the data supplied by the input patterns themselves.

2.2 Physical details of the net.

The network investigated here consists of 36 3-input s.l.a.m.s connected to a 6x6 matrix at input and output, as shown in Figure 2.1.

Notation:

Input matrix $= [I] = \{i_1, i_2, \text{----}\ i_n\}$ $1 \leqslant n \leqslant 36$

Output matrix $= [O] = \{o_1, o_2, \text{----}\ o_n\}$ $1 \leqslant n \leqslant 36$

"Teach" matrix $= [T] = \{t_1, t_2, \text{----}\ t_n\}$ $1 \leqslant n \leqslant 36$

Input to s.l.a.m.s $= [X] = \{x_{11}, x_{21}, \text{----} x_{jk}\}$ $1 \leqslant j \leqslant 3$

$1 \leqslant k \leqslant 36$

Random connections are made between the [I] matrix and the [X] matrix in such a way that each element of I is connected to three different s.l.a.m.s (i.e. for three inputs $x_{j_1k_1}$, $x_{j_2k_2}$, $x_{j_3k_3}$ connected to any element of

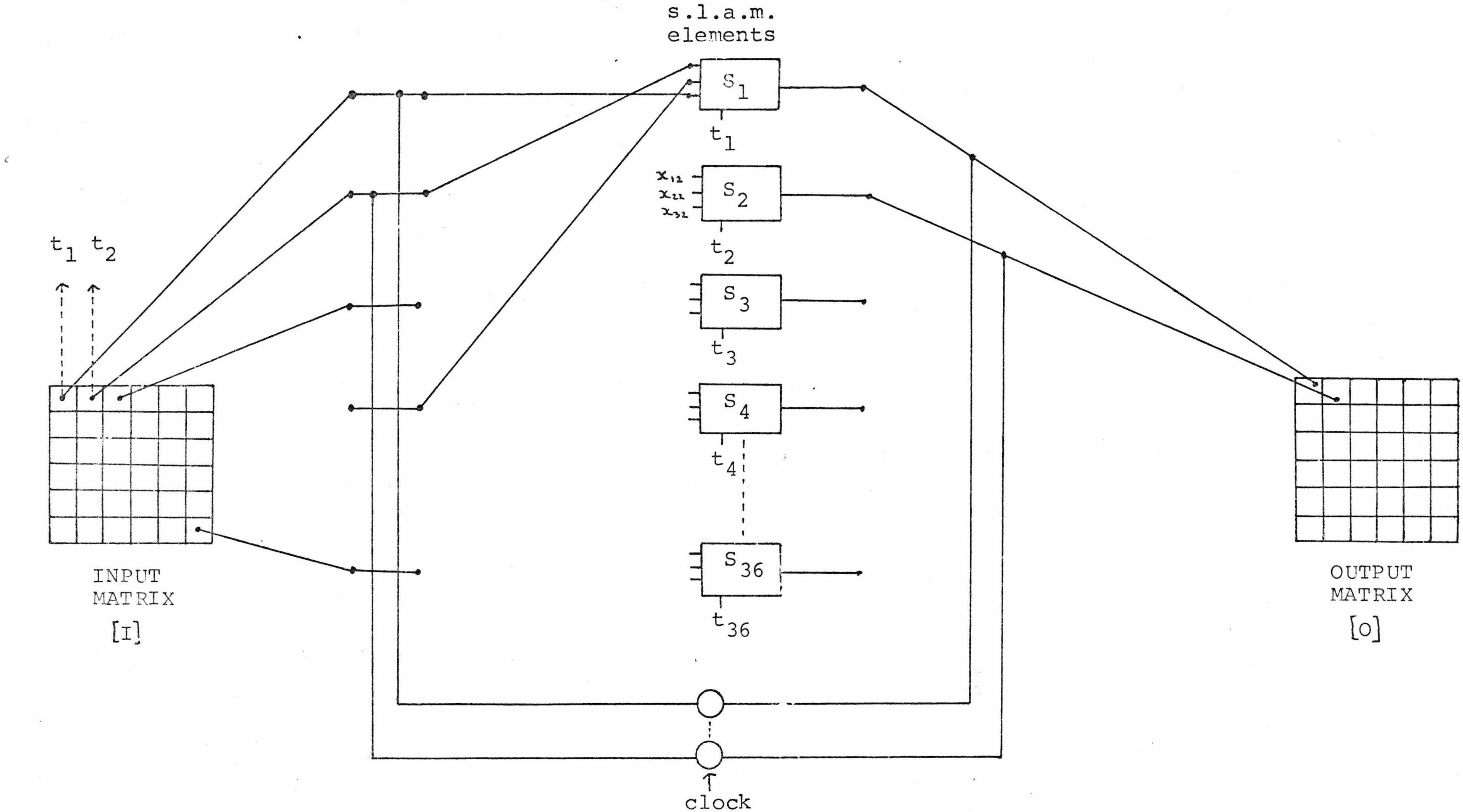


Figure 2.1: *Block diagram of the system.*

I , then $k_1 \neq k_2 \neq k_3$).

The "teach" inputs are also made to sample the input to the net, such that $t_n = i_n$ for all values of n, and the [O] matrix is connected in such a way that an output o_n is forced by the teach signal t_n (for n=1 to 36) during the teaching phase.

The net is dynamic in the sense that an input (stimulus) produces a response which is fed back to become the next input to the net. The net is assumed to cycle in discrete intervals of time (i.e. controlled by a clock), and the feedback connections are such that the output element o_n at a time θ becomes the next input element i'_n at time $(\theta+1)$, for all values of n. This is the method of connection first introduced by Aleksander and Mamdani (1968)[74].

2.3 Experimental patterns.

It is necessary to make some comment on the patterns used in the following experiments with the net described, since, in order to make any objective observations about the performance of the net, some quantitative parameters of the experimental patterns must be known.

Data for experiments in pattern recognition usually consists of samples taken from the pattern classes to be recognised. An obvious example is that of taking samples of letters and numbers written by many different people (see e.g. Neisser and Weene, 1960[75]), these samples corresponding to noisy, distorted or translated versions of what can be called "prototypes" of the individual classes.

In a similar way the pattern classes used here are formed by defining a set of generator patterns (prototypes of the required classes) from which patterns at various Hamming distances can be derived. The Hamming distance between two patterns is simply the number of binary points in which they differ, and as such is a convenient measure of their similarity.

For instance, consider two patterns P_1 and P_2, represented by the vectors

$$[P_1] = \{p_{11}, p_{12}, \text{----} \; p_{1n}\}$$

$$[P_2] = \{p_{21}, p_{22}, \text{----} \; p_{2n}\}$$

Performing an "exclusive OR" operation between the corresponding elements of $[P_1]$ and $[P_2]$ a new vector

$$[D] = \{d_1, d_2, \text{----} \; d_n\}$$

$$= \{(p_{11} \oplus p_{21}), (p_{12} \oplus p_{22}), \text{---} \; (p_{1n} \oplus p_{2n})\}$$

is formed.

Then the Hamming distance (H) between the patterns P_1 and P_2 is given by

$$H = \sum_{j=1}^{n} d_j$$

where $\sum$ refers to an arithmetical summation of logical ones.

Since most of the experimental work is concerned with two-class learning, two generators were used. These are seen (Figure 2.2) to be prototypes of the letters 'T' and 'H'. The two pattern classes to be considered therefore corresponded to variants of these two prototypes. The amount of degradation (the Hamming distance measured with respect to the appropriate generator) has possible values between 0 (zero, the generator itself) and 36 (the logical inverse of the generator). Training sets were made up of patterns at a Hamming distance of two from the respective generators except where otherwise specified.

2.4 Simplified "model" network.

Consider the network shown in Figure 2.3 which consists of four 2-input s.l.a.m. elements.

(a) (b)

Figure 2.2

(a) Generator for T-patterns.

(b) Generator for H-patterns.

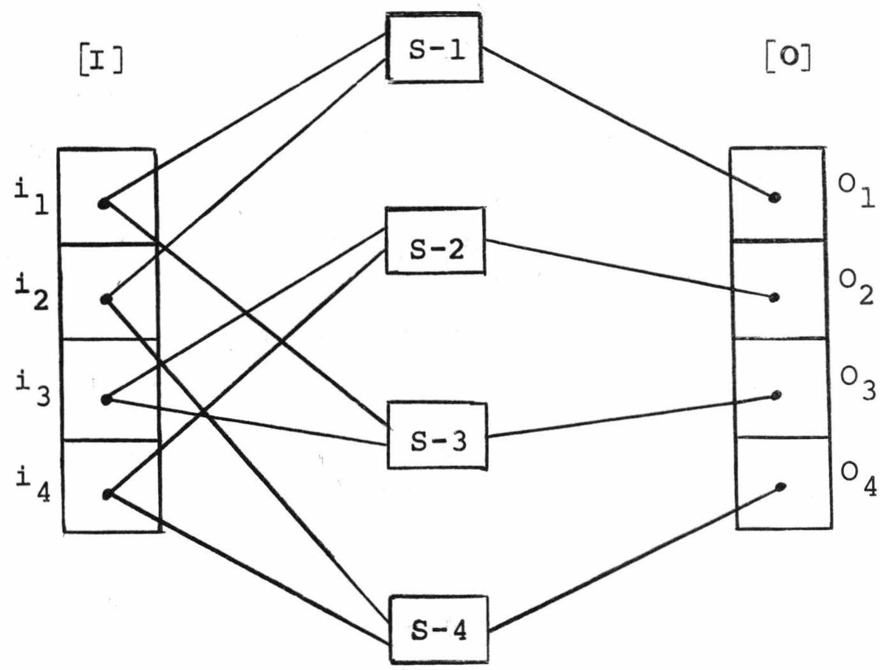


Figure 2.3 *Simple model net.*

The [I] matrix is covered twice by the s.l.a.m. inputs to give a 1:1 correspondence between input and output. The teach connections (not shown) are made as for the net described in section 2.2, so that each t_n is connected to the corresponding i_n.

Since this net has only sixteen possible states its entire state space structure can easily be represented and its behaviour precisely evaluated. In this respect it provides a useful model which will be used as a basis for the explanation of the observed behaviour in larger practical systems.

2.5 One-class learning.

2.5 (i) Empirical results.

The net was trained on ten patterns of the same class with the teach terminals of the s.l.a.m.s sampling the input matrix completely as described in section 2.2. The teach signals were then cut off (i.e. the "teach phase" terminated and the learnt logic functions of the elements fixed) and the state sequence observed for a set of test patterns which included the patterns of the training set. The state of the system at any time is simply the output vector [O]. The feedback path around the system causes a sequence of outputs which eventually reaches a stable state or a recurring cycle of states.

The resulting state space behaviour is shown in Figure 2.4, were states are plotted with respect to their Hamming distance from the last-seen training pattern (this is labelled P). All states therefore lie on a set of concentric circles radiating from P, whose radii represent the Hamming distance from pattern P. The transitions are also drawn to scale with respect to Hamming distance between successive states.

The pattern which was "seen" last by the net in its period of training is obviously dominating the subsequent performance of the net, since the state sequences of all other training patterns are converging around it. This dependence on the order of training and the influence of the last pattern in a training sequence is discussed further

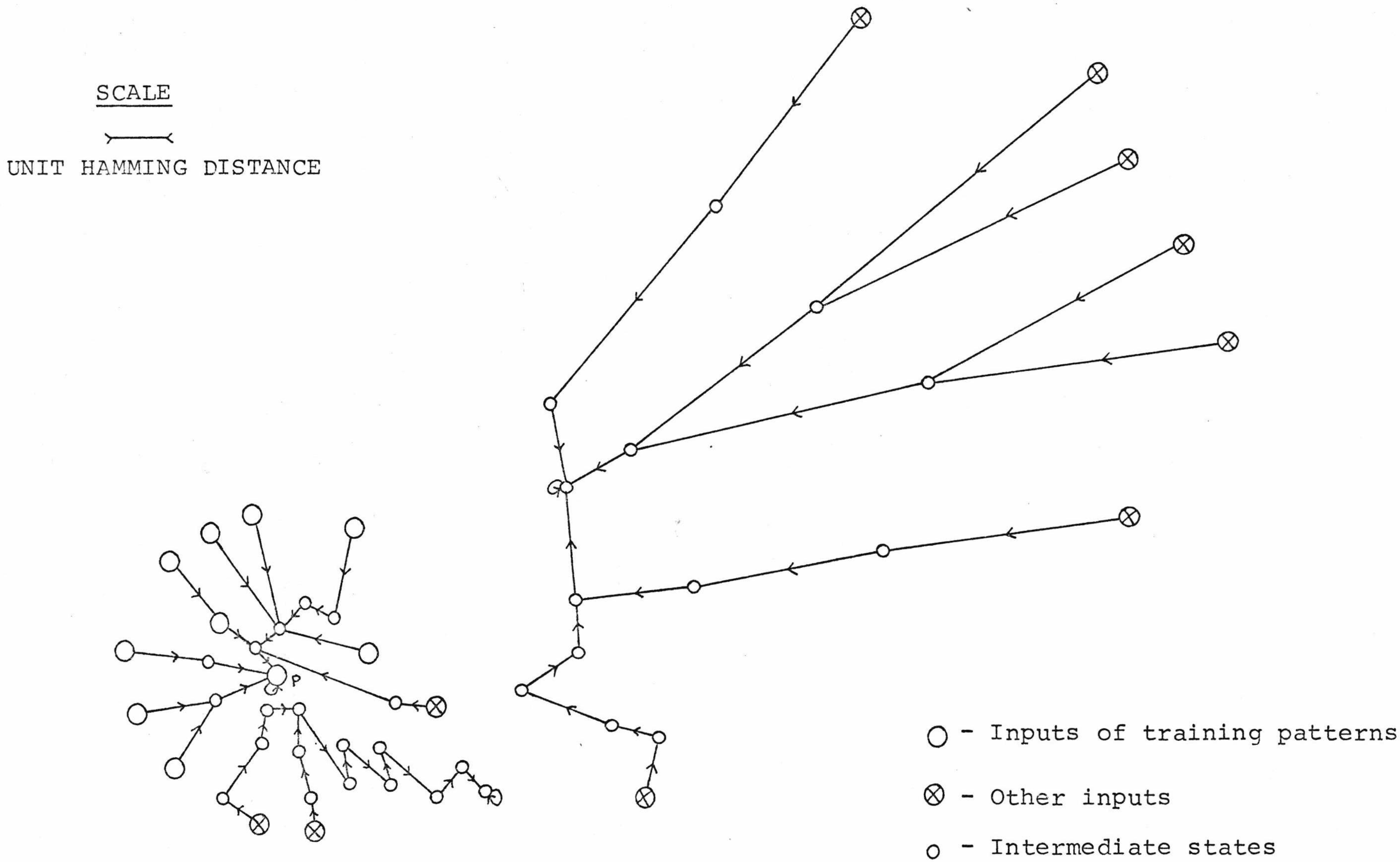


Figure 2.4: State space behaviour in one-class learning.

in section 2.8.

Figure 2.4 also shows how all patterns, including those which begin at a large Hamming distance from the training set - and could therefore belong to a completely different class - are drawn in towards the training patterns.

It appears that here the net forms an internal model of the environment, which in this case consists of only one pattern class. An explanation of the convergence of states around the last-seen training pattern is best provided by referring to the model net described in section 2.4.

2.5 (ii) One-class learning in the model net.

In order to gain an understanding of the observed behaviour, it is instructive to consider some instances of one-class learning in the model net.

When untrained, with all stores set to zero, this net has the very simple state diagram shown in Figure 2.5.

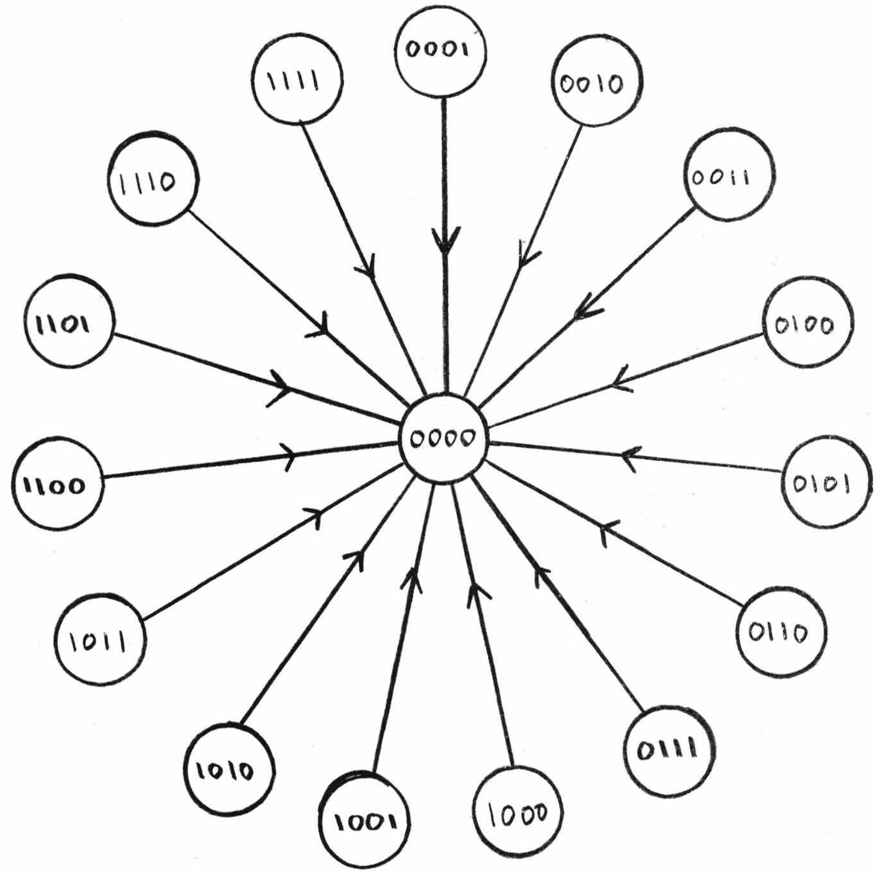


Figure 2.5 *State space of the untrained model net.*

Now consider, step by step, training this net on a series of similar patterns.

(a) Train on (0111)

On training with a single pattern (0111), the state diagram is modified as shown in Figure 2.6(a). Because of the connection of the teach wires, the training pattern obviously forms a stable state (0111). The chain leading

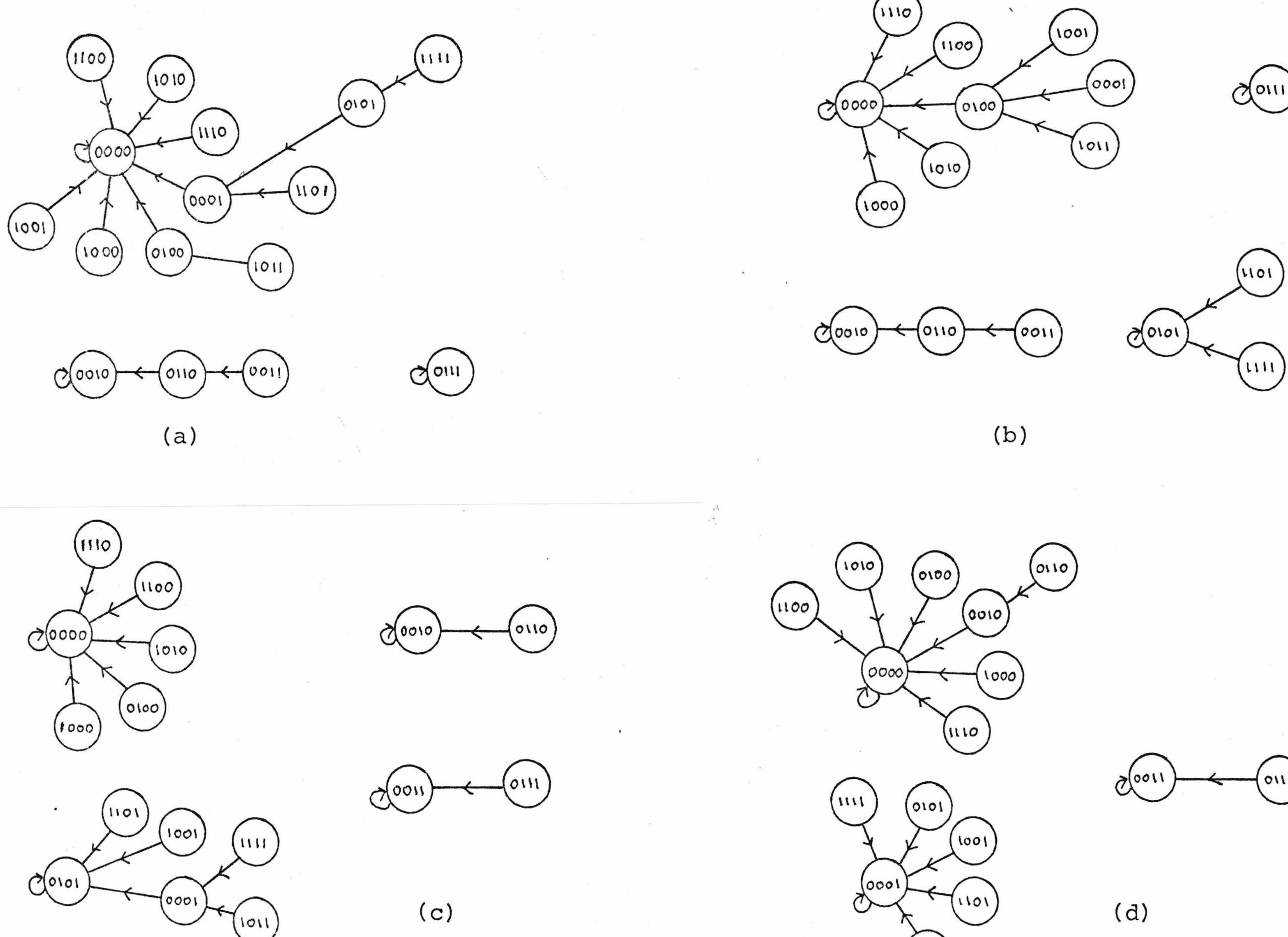


Figure 2.6: *Changes in state diagram during one-class learning.*

into (0010) is formed because input (0010) gives (01) at the input to element S-3*, as does (0111), and therefore causes an output of 1 at S-3 which sustains pattern (0010). Also (0110) addresses the location in S-3 which contains a 1, which is not the case for the other elements. This causes the transition (0110)→(0010). (0011) addresses S-2 and S-3 in the same way as the training pattern (0111), causing them to output 1's and produce the transition (0011)→(0110). (0000) is a recurring stable state and is entered by many other states, since no addresses in common with (0111) which require storage of a 1 occur.

In fact, all changes in the structure of the state diagram are due to common addresses between the states involved in the change and the pattern (0111) which was "seen" by the net during training.

(b) Add another similar training pattern so that the training sequence becomes (0111), (0101). The resulting state space diagram is shown in Figure 2.6(b).

Here the new training pattern (0101) forms its own small cluster. This drawing in of other states can be explained as follows:

---

*See Figure 2.3, page 33.

If the two training patterns are

$$\begin{bmatrix}0\\1\\1\\1\end{bmatrix} \quad \text{and} \quad \begin{bmatrix}0\\1\\0\\1\end{bmatrix}$$

then, because of the net connections,

1. O_1 can never be 1.

2. O_2 is 1 for inputs (01), (11) to S-2.

3. O_3 is 1 for input (01) to S-3.

4. O_4 is 1 for input (11) to S-4.

In other words, for the output [O] to be

$$\begin{bmatrix}0\\1\\0\\1\end{bmatrix}$$

the input matrix [I] must be

$$\begin{bmatrix} d \\ 1 \\ d \\ 1 \end{bmatrix}$$

where

d = "don't care"

Hence, states which could form a (0101) cluster are:-

(0101) (0111) (1101) (1111)

However, any input which has the feature

$$\begin{bmatrix} 0 \\ \ \\ 1 \\ \ \end{bmatrix}$$

must be excluded, since this would make $O_3 = 1$.

The states in the (0101) cluster are therefore

(0101) (1101) (1111)

The other training pattern (0111) is not affected.

(c) Training sequence (0111), (0101), (0011) [see Figure 2.6(c).]

Here (0111), which was an early training pattern, is drawn into the latest training pattern (0011). This is because O_2 is now forced to logical 0 for input (11) to S-2, whereas previously S-2 was taught to output 1 for this input. A "contradiction" in training has occurred, a later address is common with that of an early pattern disturbing its ~~neat~~ next state mapping.

Further states are drawn into (0101) as follows:

1. O_1 cannot be 1.

2. O_2 is 1 for input (01) to S-2.

3. O_3 is 1 for input (01) to S-3.

4. O_4 is 1 for inputs (01) (11) to S-4.

For [O] to be $\begin{bmatrix}0\\1\\0\\1\end{bmatrix}$ the input [I] must be $\begin{bmatrix}d\\d\\0\\1\end{bmatrix}$

and so (0001), (0101), (1001), (1101) from a cluster. States (1111) and (1011) are drawn in by virtue of their entering state (0001).

(d) Finally, make the training sequence

(0111), (0101), (0011), (0001) [see Figure 2.6(d)].

Here again a "contradiction" occurs (S-2 is forced to output 0 for (01) at the input). Once more the last-taught pattern (0001) is undistorted, the earlier pattern (0101) being distorted and drawn in to (0001).

There are, therefore, several factors which cause states to be drawn in towards the training patterns, and particularly to the last-seen pattern:

<u>The generalising property</u> of s.l.a.m. nets(due as it is to common addressing)means that an output forced during

training can be produced by previously unseen patterns. The system here is, in effect, two separate single-layer nets (Net 1 = S-1, S-2, Net 2 = S-3, S-4), each making a complete sample of the input. Since each net has generalising effect, the probability of a considerable number of patterns being drawn in to a particular state is high. In the practical system of section 2.2 the input matrix is covered three times.

Contradiction during training. If for two training patterns a different output is forced for the same input to any particular element, then the earlier pattern is distorted and the forced output of the last-seen pattern is dominant. The probability of contradiction increases as the training set becomes larger. The 'drawing-in' around the last-seen training pattern therefore becomes more marked.

2.6 Two-class learning.

The natural progression from the experiment described in section 2.5(i) is to consider the case where the system is trained on two pattern classes.

Using the same "teach" connections as before the large net was trained on ten patterns, five from each of the two training groups (patterns at Hamming distance of two from the T-generator or the H-generator). Because of the importance of the order of training, patterns of the two groups were presented to the net alternately. After training was completed, the response of the net was observed for a range of patterns including the training set.

Typical state space behaviour is illustrated in Figure 2.7 where each state is located in terms of its Hamming distance from the two generator patterns (labelled T and H respectively).

This diagram shows how the state sequences initiated by inputs from the two separate classes eventually merge into a common recurring cycle in state space. The T and H patterns also enter other cycles in this "indeterminate" region of state space (i.e. not particularly close in Hamming distance to either generator), and similarly show no correspondence to one input class in preference to the other. This type of behaviour was found not only for inputs in the immediate neighbourhood of the generator, but occurred also for stimulus patterns in regions of state space at larger Hamming distances. Of the fifty inputs tested only five separate cycles were found, two of these consisting of a single state corresponding to the last-seen training pattern from the T-group and H-group respectively.

In order to avoid confusion only two typical state space trajectories are shown in Figure 2.7.

It is clear that despite having been trained on the

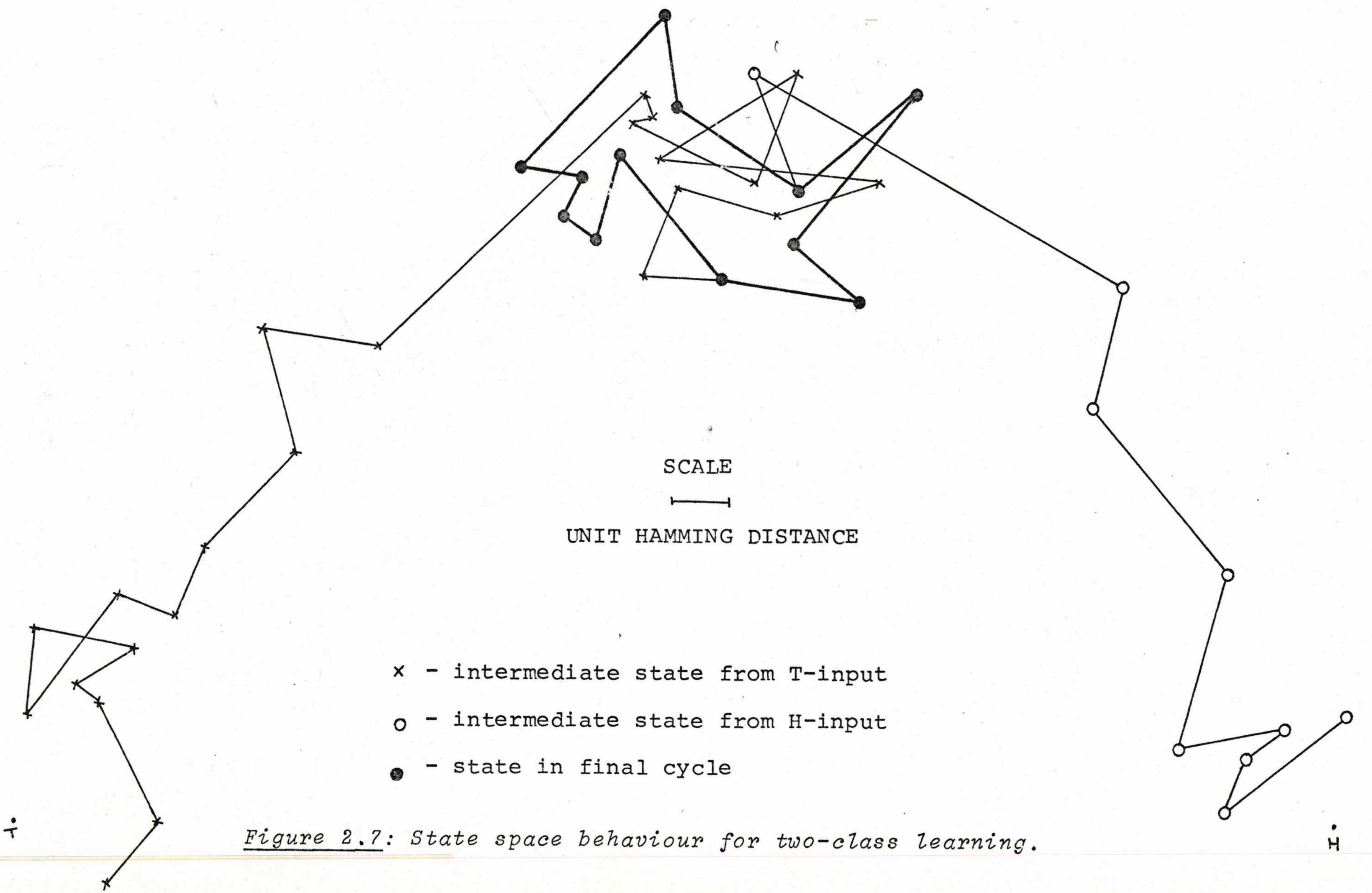


Figure 2.7: State space behaviour for two-class learning.

two distinct pattern classes, the net is unable to differentiate Ts and Hs. Some insight into this behaviour can be gained by considering two-class learning in the model net of section 2.4.

For this net, two training groups can be defined such that

<u>Group A</u> consists of (0101), (0111)

<u>Group B</u> consists of (1010), (1000).

When the net is trained on the sequence (0101), (1010), (0111), (1000) it has the state diagram of Figure 2.8. Here the structure is much more "diffuse" than in the case of one-class learning. (There is much less drawing together of states).

Consider the connections and the training patterns

1. O_1 will be 1 for input (01) to S-1,

2. O_2 will be 1 for inputs (01), (11) to S-2,

3. O_3 will be 1 for inputs (11), (01) to S-3.

4. O_4 will be 1 for inputs (01), (11) to S-4.

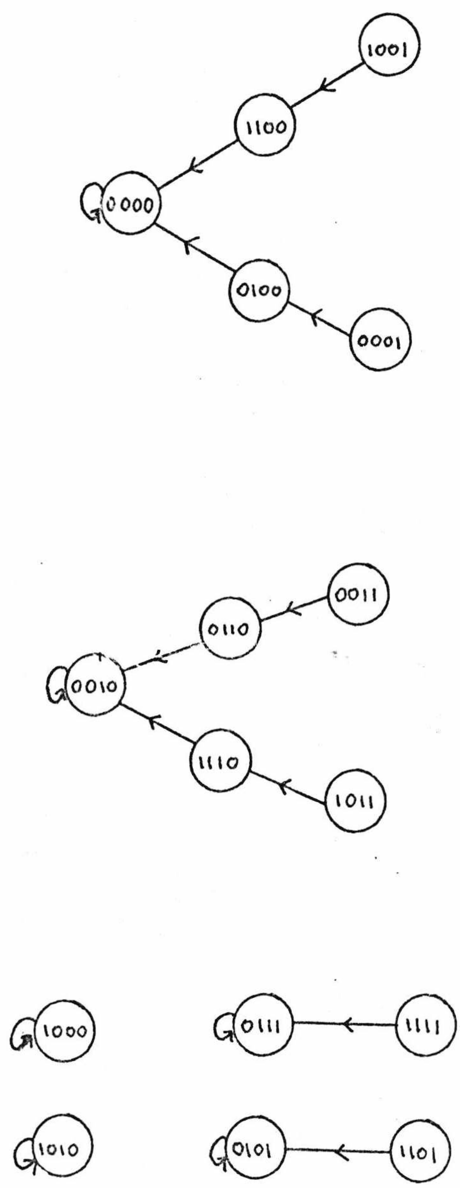


Figure 2.8: *Two-class learning in model net.*

Here it is seen that there are more <u>trained</u> locations than in the previous case. Thus more stringent conditions are imposed on the input matrix requirements for the drawing in of states to the training set. On larger nets, since fewer states are absorbed into the training patterns, the output sequence tends to disintegrate and drift into some indeterminate area of state space.

The experiments described in the next chapter will investigate the question of applying constraints to the system such that the state space activity is confined to distinct regions associated with patterns of the two classes.

<u>2.7 Initial conditions</u>.

The experiments described in section 2.5 and section 2.6 previously were carried out with the stores of individual s.l.a.m. elements reset (to logical zeros) before training began. Under these conditions only those store locations actually addressed during training have the possibility (but not the certainty, due to the nature of the teach patterns) of containing a logical one. As a result there is a tendency for previously unseen inputs to produce output states containing a large number of zeros.

This starting point can be responsible for the behaviour of the system, and it is interesting to study the system with s.l.a.m. stores set initially at random. This is achieved by allocating random logic functions (such that logical ones and logical zeros occur with approximately equal probabilities) to the s.l.a.m. elements before training is begun. In subsequent use of the net, addressing of locations which were not specifically trained to a desired output will therefore cause an output which has an equal probability of being a one or a zero.

These initial conditions result in there being greater dissimilarity in the output states (due to less domination by output zeros). The overall effect is a tendency for output sequences to become longer than in the corresponding case with stores initially cleared. Typically, sequence lengths increased from twenty states to approximately fifty states, the effect being less marked for T-patterns than for H-patterns.

For a net with stores initially reset the response to <u>any</u> input before training will be an output vector [O] whose individual elements are all zero. For a net with initially random stores the state sequences before training form cycles in state space such as the one shown in Figure 2.9. (This was the only cycle found for forty inputs tested).

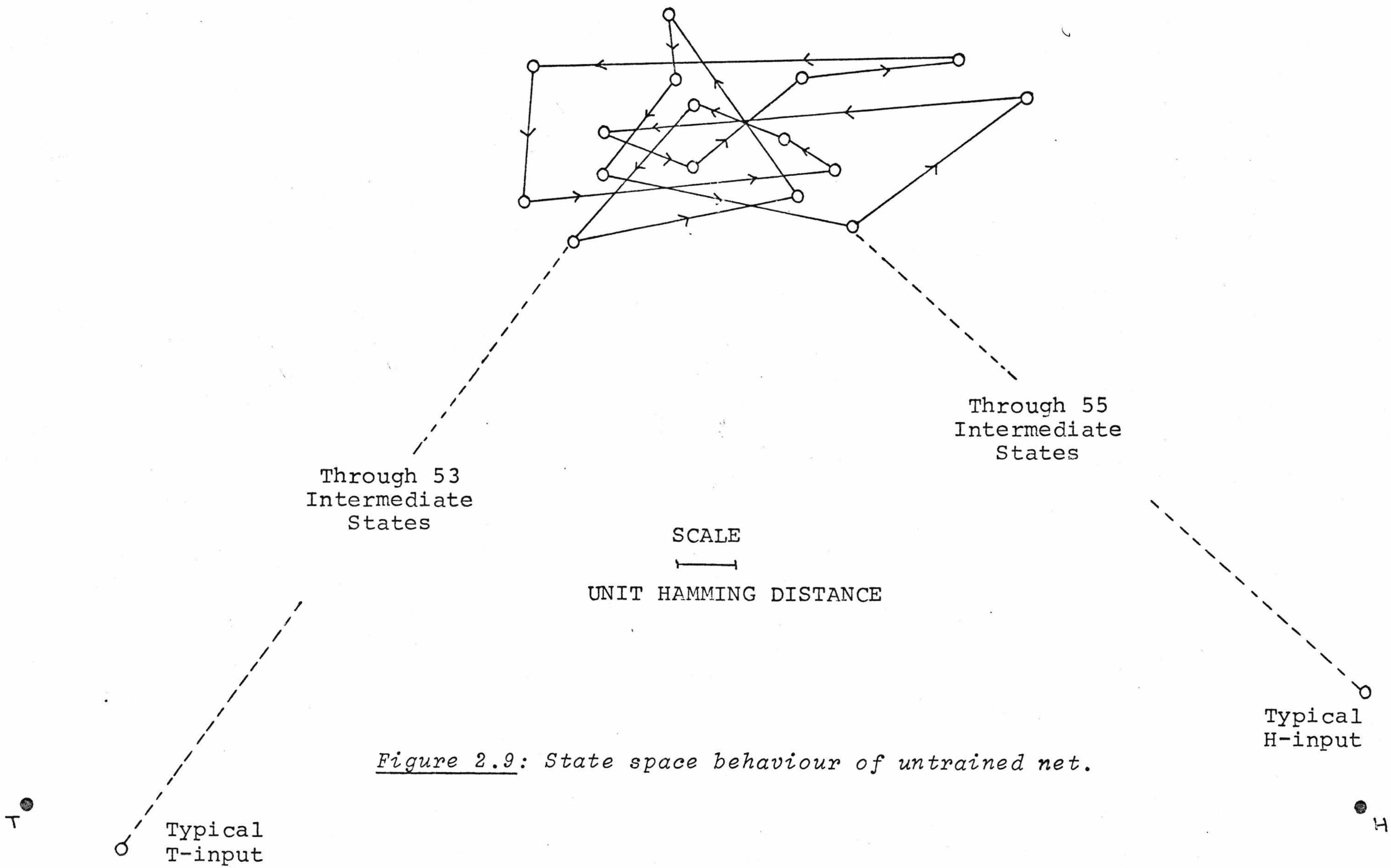


Figure 2.9: State space behaviour of untrained net.

In all future experiments initially random stores will be assumed unless other conditions are specifically stated.

2.8 Training sequence dependence.

It was stated in section 2.5 that when the "teach" terminals of the s.l.a.m.s are made to sample the input pattern, then the order in which a training set is presented to the net can affect its subsequent performance. This can easily be understood in terms of the generalisation and contradiction effects discussed in section 2.5(ii). This section provides some practical evidence for the importance of order of training, which can be illustrated as follows.

A training set of five patterns was chosen, each pattern at a Hamming distance of two from the generator.

A test set consisted of five patterns A, B, C, D, E, each with a different Hamming distance with respect to the generator. These values are given in Table 2.1.

Test pattern	Hamming distance from Generator
A	2
B	6
C	10
D	22
E	28

Table 2.1 *Hamming distance values for test patterns.*

The net was trained using the teach connections described in section 2.2. The test patterns were then shown the net in turn and in each case the output observed and its Hamming distance from the generator measured. (For simplicity, no feedback was allowed during these tests). The s.l.a.m. stores were then reset and the net trained again, this time with the same five training patterns, but presented in a different order. The response to the test set was again observed.

The whole procedure was repeated ten times so that the variation in response as a function of order of training could be measured. The distributions for the five test patterns are shown in Figure 2.10.

The results show that a net trained in this way is sensitive to order of training, particularly when a test pattern is similar to the patterns seen in training (e.g. test patterns A and B). This is investigated theoretically in Chapter 6.

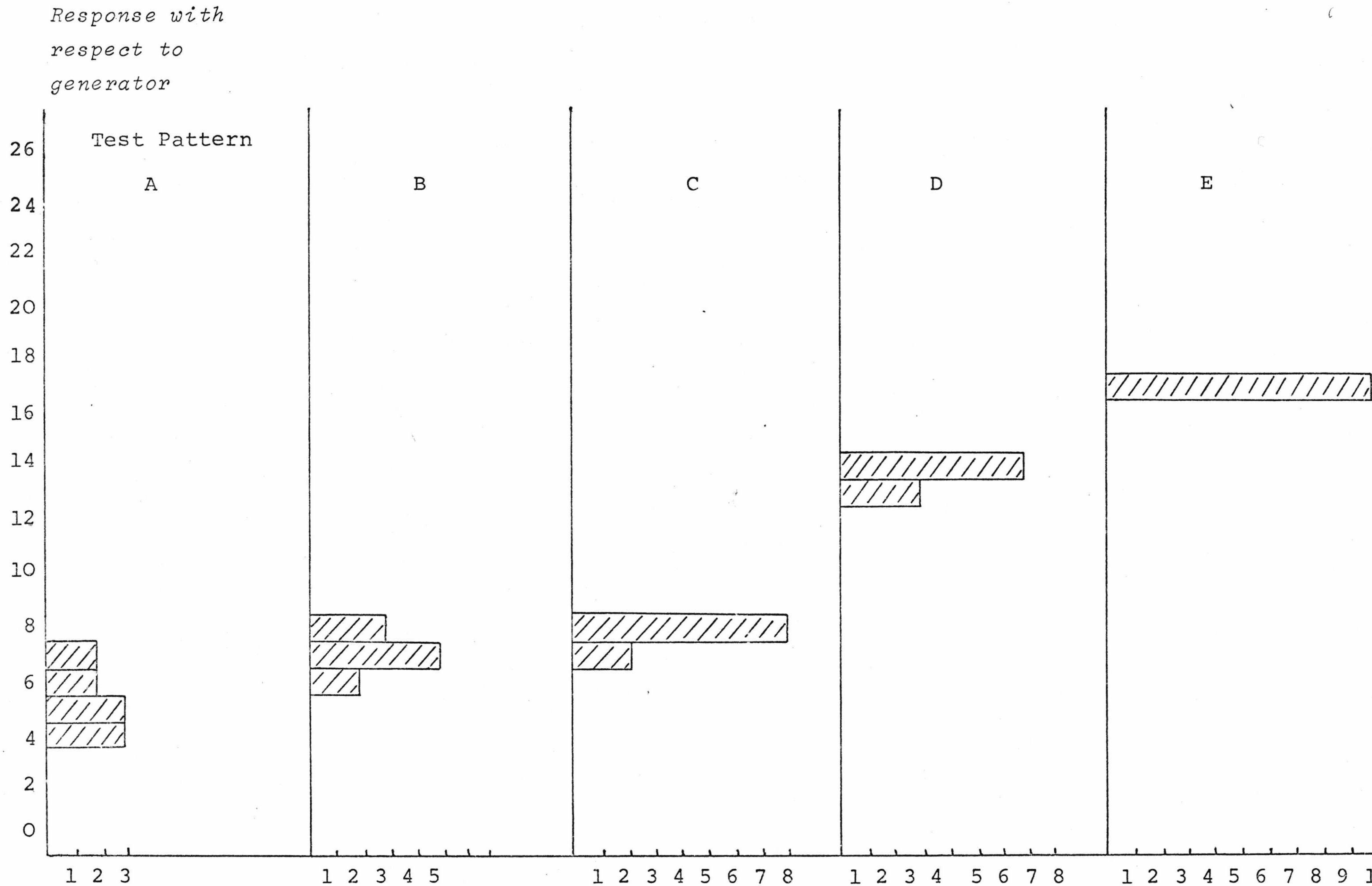


Figure 2.10: Response distributions as order of training varies.

CHAPTER 3

PATTERN CLASSIFICATION BY A CONSTRAINED NETWORK

3.1 Constraints on the system.

Chapter 2 considered an unsupervised network of s.l.a.m. elements where the overall input-output logic function was changed as the elements received teach signals derived from an applied stimulus pattern. As a result of the learning process the state space structure became reorganised in such a way that the possible areas of activity were undefined in terms of the pattern classes seen by the net.

From a practical point of view it is desirable to find a way of constraining the system so that, by controlling its attainable state sequences, pattern classes present in the learning environment may be identified with definite state space domains. One way of doing this is to include a set of OR gates between the applied stimulus and the pattern in the feedback path, thereby maintaining the influence of the input during successive cycles of the net.

The arrangement is shown in Figure 3.1. The logical OR operation between the stimulus pattern and the feedback pattern means that the actual pattern sampled by the s.l.a.m. inputs is a new matrix

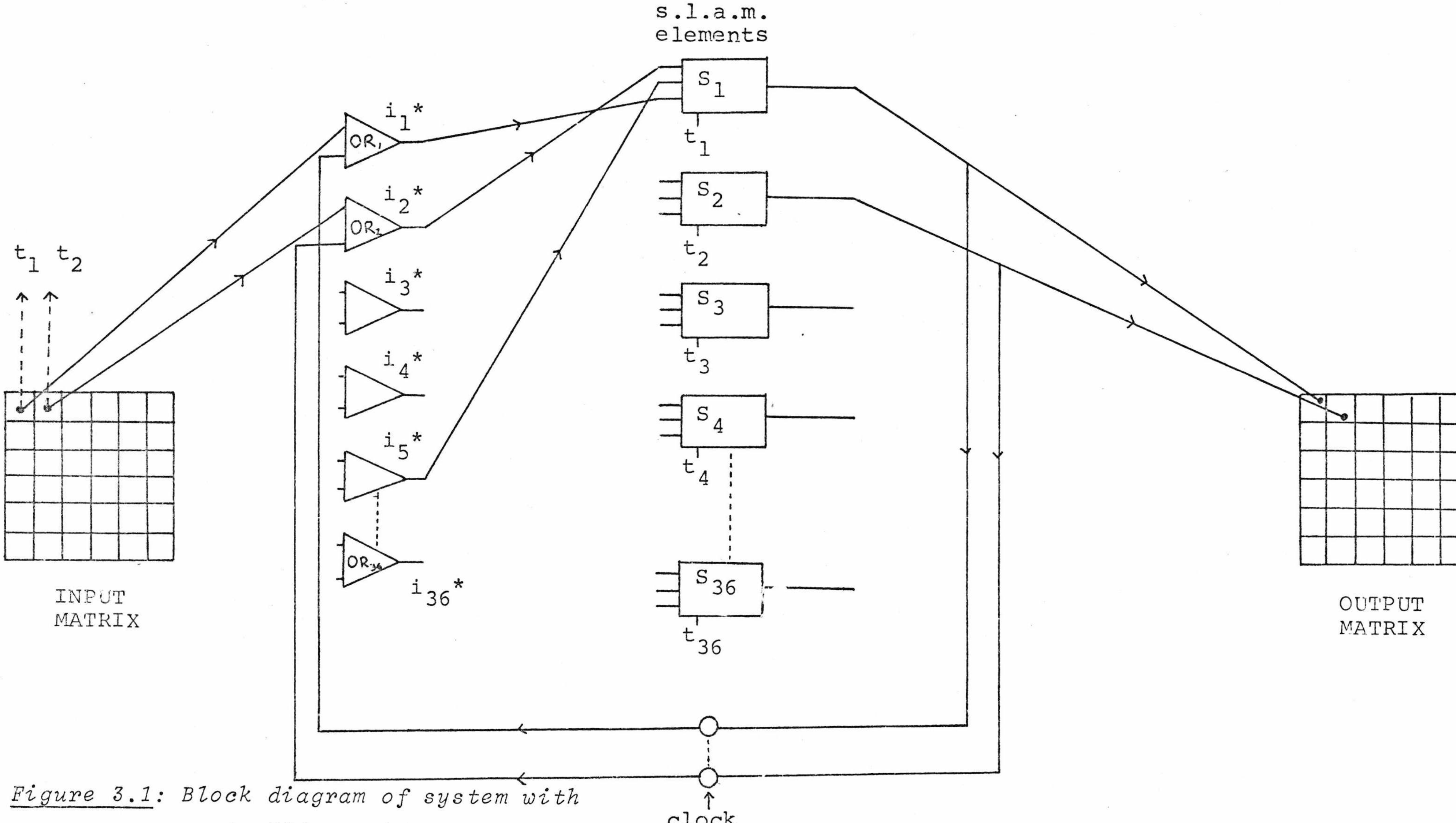


Figure 3.1: Block diagram of system with controlling gates.

$$[I^*] = \{i_1^*, i_2^*, \text{----}\ i_n^*\}$$

which is related to [I] by

$$i_n^* = i_n + o_n' \quad \text{(for all n)}$$

where o_n' is the n^{th} element of [O] of the previous cycle. The nature of the OR function and its effect on the state dynamics of the net is discussed in the next section.

3.2 Dynamics of a network with OR-ed feedback.

The net was trained on five patterns of each group as in section 2.6, and tested with a series of inputs, but this time with the stimulus [I] held at the input and OR-ed with the pattern in the feedback path to produce a new input [I*].

The OR function provides a means by which the stimulus to the net can control the subsequent state sequence. Figure 3.2 shows how the net is driven into areas of state space in which identity with distinct pattern classes can be established. T-inputs and H-inputs close to the generators no longer give rise to common cycles, but produce short cycles or recurring single states which are quite distinct.

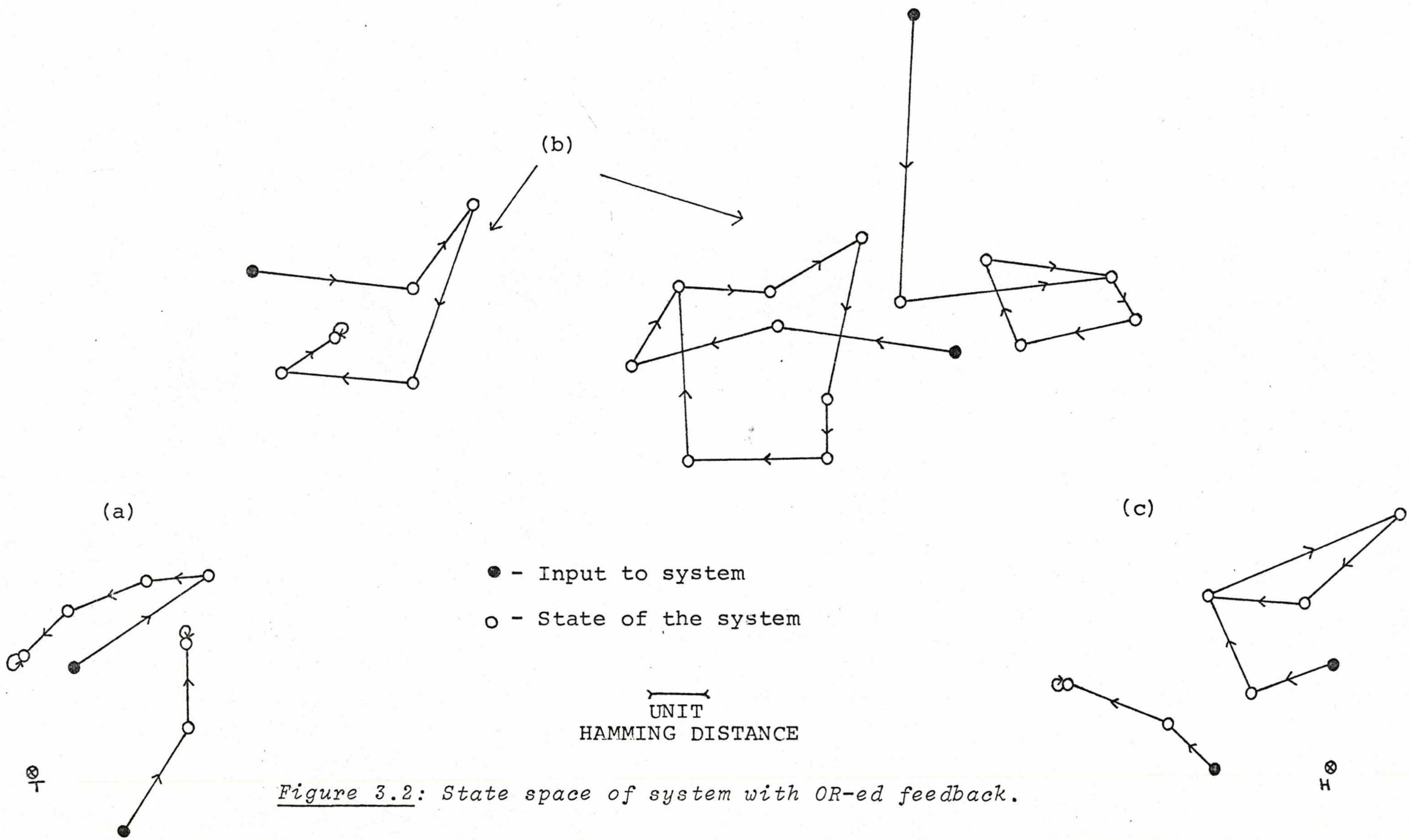


Figure 3.2: State space of system with OR-ed feedback.

In all, fifty inputs were tested. Inputs of the two training classes gave rise to stable states or short cycles in the region of their respective generator (as in Figure 3.2(a) and Figure 3.2(c)), while "indeterminate" inputs generally gave rise to cycles in indeterminate state space (Figure 3.2(b)).

i_n	o_n'	i_n^*
0	0	0
0	1	1
1	0	1
1	1	1

Table 3.1: *Truth table for i_n^**

The OR gates at the input of the system have the effect of "locking out" certain areas in state space. As a result of the nature of the OR function, any element of [I] which is a logical 1 will always cause a logical 1 in the corresponding element of [I*]. Only when i_n is 0 and o_n' (the previous output) is 1 can an element of [I*] change (see Table 3.1). The number of elements of [I*] which can change with successive passes through the net is therefore considerably restricted.

Since the input matrix to the s.l.a.m.s, [I*], is partially fixed, only a limited number of elements can contribute to the change in the output vector [O] as the net cycles.

The restricted activity of the net has two effects.

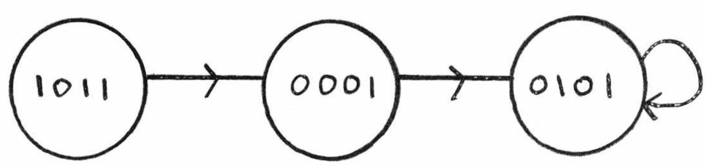
(a) Since effectively fewer elements are active, the likelihood of any particular element of [O] changing in successive outputs is reduced, as a result of which there is generally a smaller Hamming distance between inputs [I*] to the s.l.a.m.s. There is, therefore, a tendency for cycles to become shorter.

(b) If a stimulus pattern is similar (in the Hamming distance sense) to patterns of the training set, then the elements which are fixed are likely to contribute to the output [O] in such a way that the output pattern is also similar to the appropriate training class.

Hence, as Figure 3.2 shows, the trained pattern classes are distinct and do not produce common cycles as with the unconstrained net. Since with OR-ed feedback the <u>structure</u> of the net is changed for each new input (different elements of [I*] are "frozen" for different inputs), the stimulus pattern, in effect, controls the state sequence of the system.

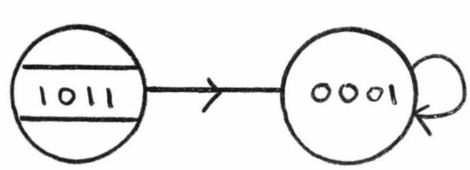
The mechanism of the OR function can be illustrated simply by reference to the model net described in section 2.4. Consider the net when trained on three patterns (0111), (0101), and (0011). It then has the state diagram previously shown in Figure 2.6(c) on page 38.

As an example, consider the input (1011). When the net is allowed unrestricted cycling, this input gives rise to the chain



as shown previously.

If the same input is presented to the net with OR-ed feedback connections it is found that the chain becomes stable after the first output. In other words the chain becomes



[The symbol 

indicates an input to the net rather than a state of the system].

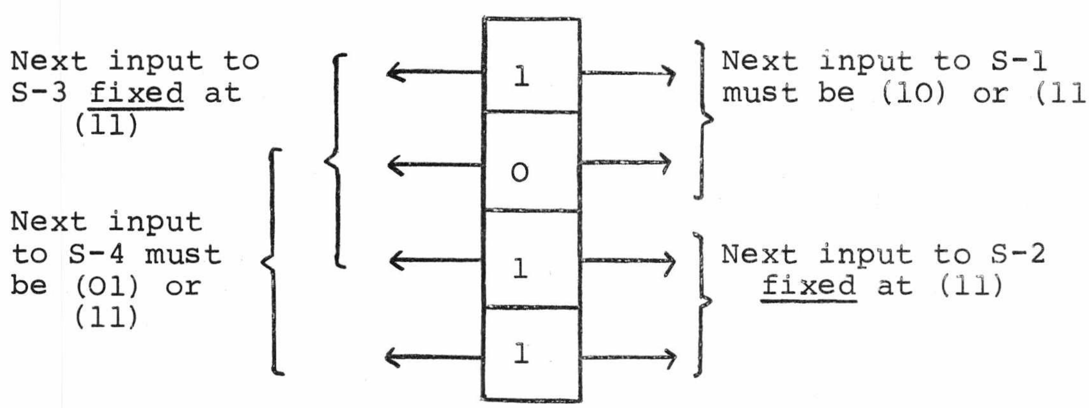
The reason for this behaviour becomes evident if we consider the nature of the OR function as shown in Table 3.1.

For the input

i_1	1
i_2	0
i_3	1
i_4	1

only i_2* can change in the next input to the net since i_1*, i_3*, and i_4* are clamped by the presence of 1s at i_1, i_3, and i_4.

Now consider the connections to the net



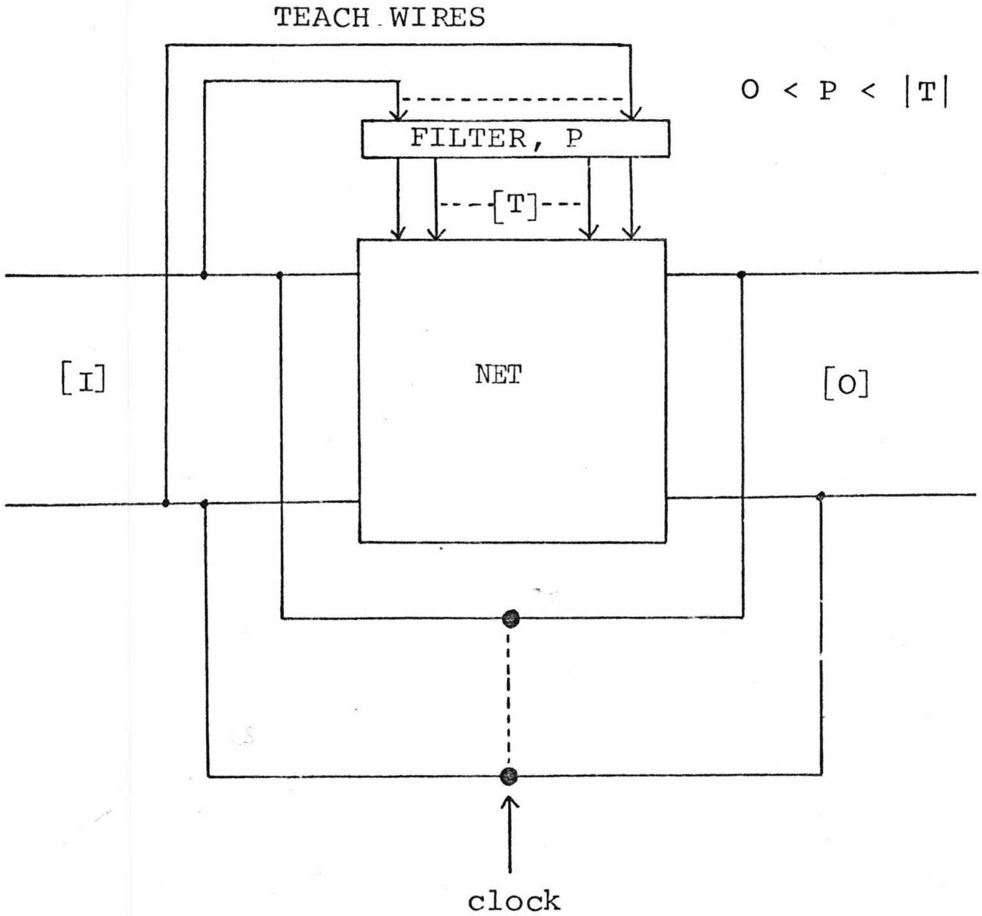
This shows how the activity of the net is restricted by the OR function. Since the inputs to element S-2 are fixed by virtue of the fact that the influence of the stimulus is maintained during cycling, the s.l.a.m.s will actually see the same input matrix [I*] at every pass through the net.

3.3 "Aging" in the teaching process.

In conventional applications of s.l.a.m. networks, two distinct modes of operation are assumed, corresponding to a "teaching" phase and a "response" phase (see e.g. Aleksander & Albrow, 1968[76]). Such an arrangement requires intervention by an external agency in order to define where the former ends and the latter begins.

An alternative approach is to allow the system to make a gradual exit from the learning phase by introducing an "aging" process into the learning period so that, as training proceeds, the net becomes progressively less sensitive to "teach" information. In practice this is realised by cutting off the teach signals from an increasing number of s.l.a.m.s so that the logic functions of more and more elements become fixed as time goes on. Eventually, after sufficient decay in activity at the teach terminals, the net will essentially have passed from the learning phase to the response phase.

The system can then be visualised as a conventional network with a filter, P, incorporated at its teach terminals (see Figure 3.3). P can assume all integer values between O (all teach signals being clocked) and $|T|$



<u>Figure 3.3:</u> Net with "filter" at teach.

(no teach signals being clocked - i.e. the response phase).

The aging process described above also serves another useful function. It has been found (Wilkins and Ford, 1972)[77] that the efficiency of a pattern recognition system can be impaired if the training set it employs contains "defective" or "unreliable" patterns (in the sense that they are poor or untypical exemplars of the pattern classes which they are supposed to represent). The aging process has the effect of making the precise characteristics of the training patterns less critical as training progresses, and hence only the very early stages of training need be strictly controlled. This is demonstrated empirically in section 3.5.

3.4 Empirical results for a net with aging.

This section presents the results of a two-class learning experiment on a network incorporating OR-ed feedback and the aging process outlined previously. The results of a small number of tests have already been published (Fairhurst & Aleksander, 1971)[78], but more comprehensive details are given here.

The network was trained on six patterns, two from each of the usual training groups (Hamming distance of 2 from the generators) and two at a larger Hamming distance

(=10) from each generator. The training set was "shown" to the net five times with the number of active teach terminals (each set a subset of the previous set) reduced at each pass through the net, as shown in Table 3.2. In each case the net was allowed to cycle until a stable output was observed.

When the net had passed into the response phase it was tested as before with a range of inputs, and the state sequences observed in each case. Figure 3.4 shows the results obtained for a sample of the patterns tested. The Hamming distances are in this case measured not from the generators, but from the most frequently occurring final state for each group (these are labelled T and H).

It appears from the diagram that a clustering operation is being performed by the net. Figure 3.4(a) and Figure 3.4(c) show the tendency for a large number of inputs to be mapped into fewer final states, which are found to be close in Hamming distance to the generators for the respective groups. It is also noticeable (in the dotted area of Figure 3.4(c)) that inputs outside the immediate neighbourhood of the training groups, but which are obviously more similar to one group than the other, are also drawn towards the appropriate cluster. Finally, in the case of inputs which are indeterminate with respect to the training groups, the system is found to enter a recurring cycle of states or a cluster of states in an indeterminate area (Figure 3.4(b)).

PASS	Number of active terminals.
1	36
2	18
3	9
4	5
5	2

Table 3.2: Decrease in teach activity.

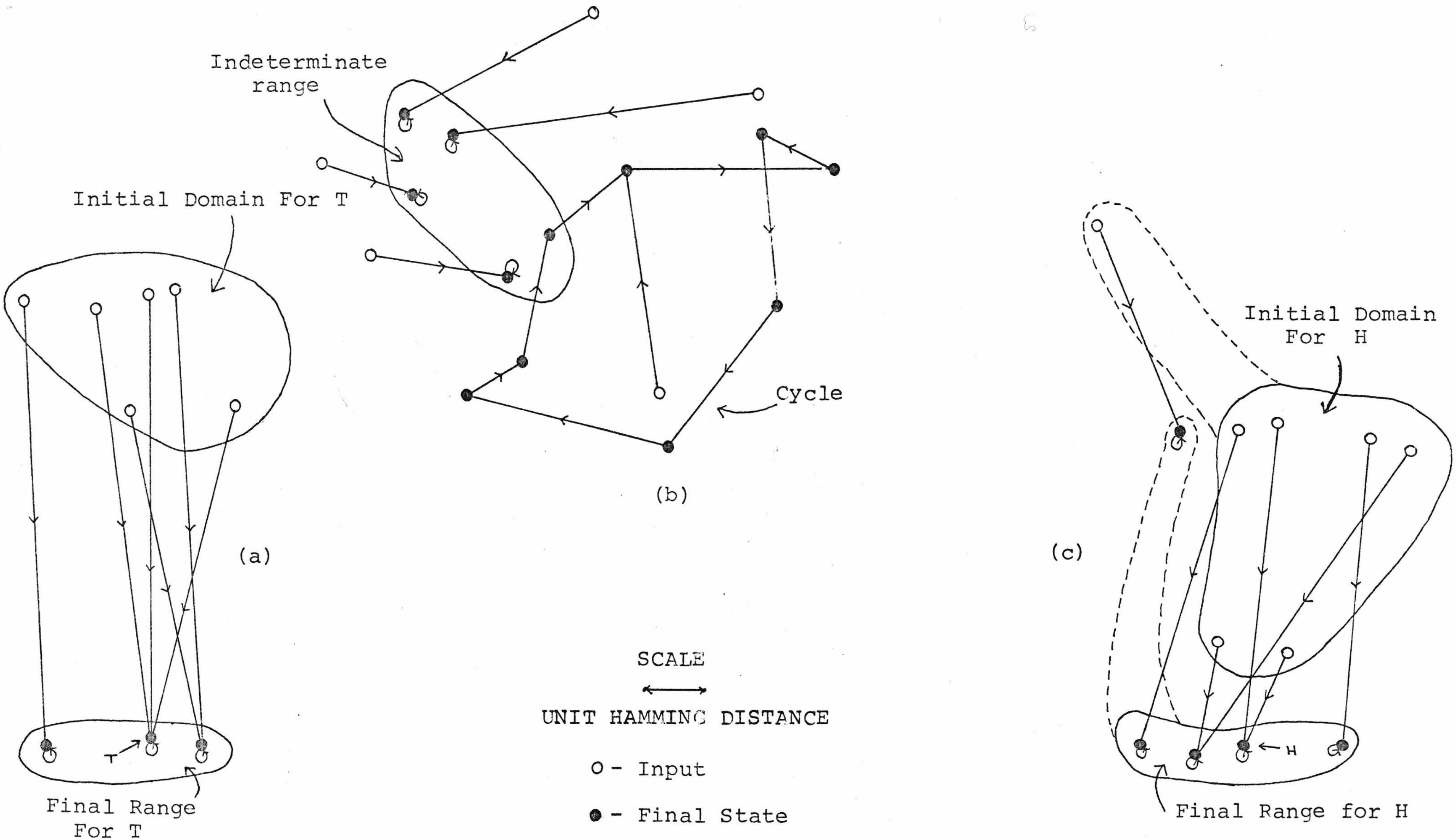


Figure 3.4: Dynamic behaviour of the net with aging.

Note that in Figure 3.4 only initial and final states are shown, although in reality the state sequences passed through intermediate states. Sequence lengths were typically found to be 3 or 4 states.

Clustering techniques are not new, having been used as long ago as 1939 (Tryon)[79], and their application to problems of pattern recognition has previously been studied (see, for instance, Bonner, 1964[80], Jardine, 1971[81]). These techniques generally involve the measurement of predetermined parameters ("attribute states") on the basis of which an input pattern can either be assigned to an existing cluster, form a new independent cluster, or cause an amalgamation of clusters. The results presented here show how the system, merely by exposure to pattern classes during a decaying learning period, performs a natural clustering operation as it internally generates "archetypes" of the training classes and draws other similar inputs into them. The archetype states (the most frequently occurring final stable states) generated for the two groups in this experiment are shown in relation to their respective generators in Figure 3.5.

In this particular experiment all the 630 possible inputs of each of the two training groups (patterns at a Hamming distance of 2 from the generators) were tested. For the 1,260 total possible inputs only 234 final states were found, the archetype states of Figure 3.5 occurring 180 and 170 times for T-inputs and H-inputs respectively. As an example, Table 3.3 shows the complete data for the

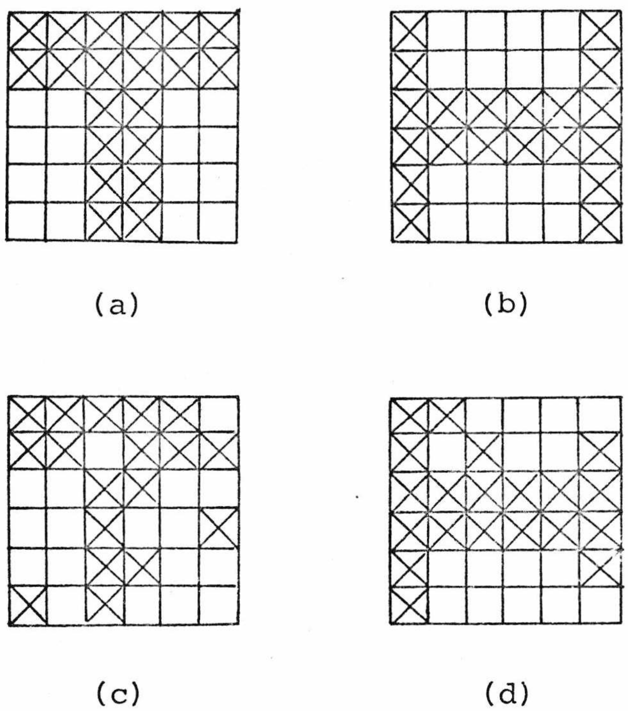


Figure 3.5:

(a) & (b) *Generator patterns.*

(c) & (d) *Corresponding archetype patterns.*

X states each occur Y times.

X	Y
1	170
1	51
2	20
5	19
2	18
1	17
1	15
1	9
8	3
2	2
60	1

Table 3.3: *Date for H-inputs.*

H-inputs.

It is significant that within each group approximately 30% of the inputs produce the same final state. Even more important is the fact that there was no ambiguity in classification - no T-inputs entered final H-states nor vice versa.

The clustering effect can be simply illustrated for a randomly selected sample of 30 T-inputs. By measuring the Hamming distance of each pattern from every other pattern for the 30 input states and similarly for the corresponding final states produced, the histograms of Figure 3.6 and Figure 3.7 are obtained. Figure 3.6 shows the predominant peak at a Hamming distance of 4 for the input states. After being processed by the net (Figure 3.7) the distribution shifts towards a Hamming distance value of zero as fewer and more similar final states occur.

Clustering in the network is found even for considerable variations in the characteristics of the training set. In particular the effect has been observed under the following conditions (Baker, 1972)[82].

(i) Using a training set consisting of patterns derived from two generators (Figure 3.8(a)) much closer in Hamming distance (=12) than in the previous case.

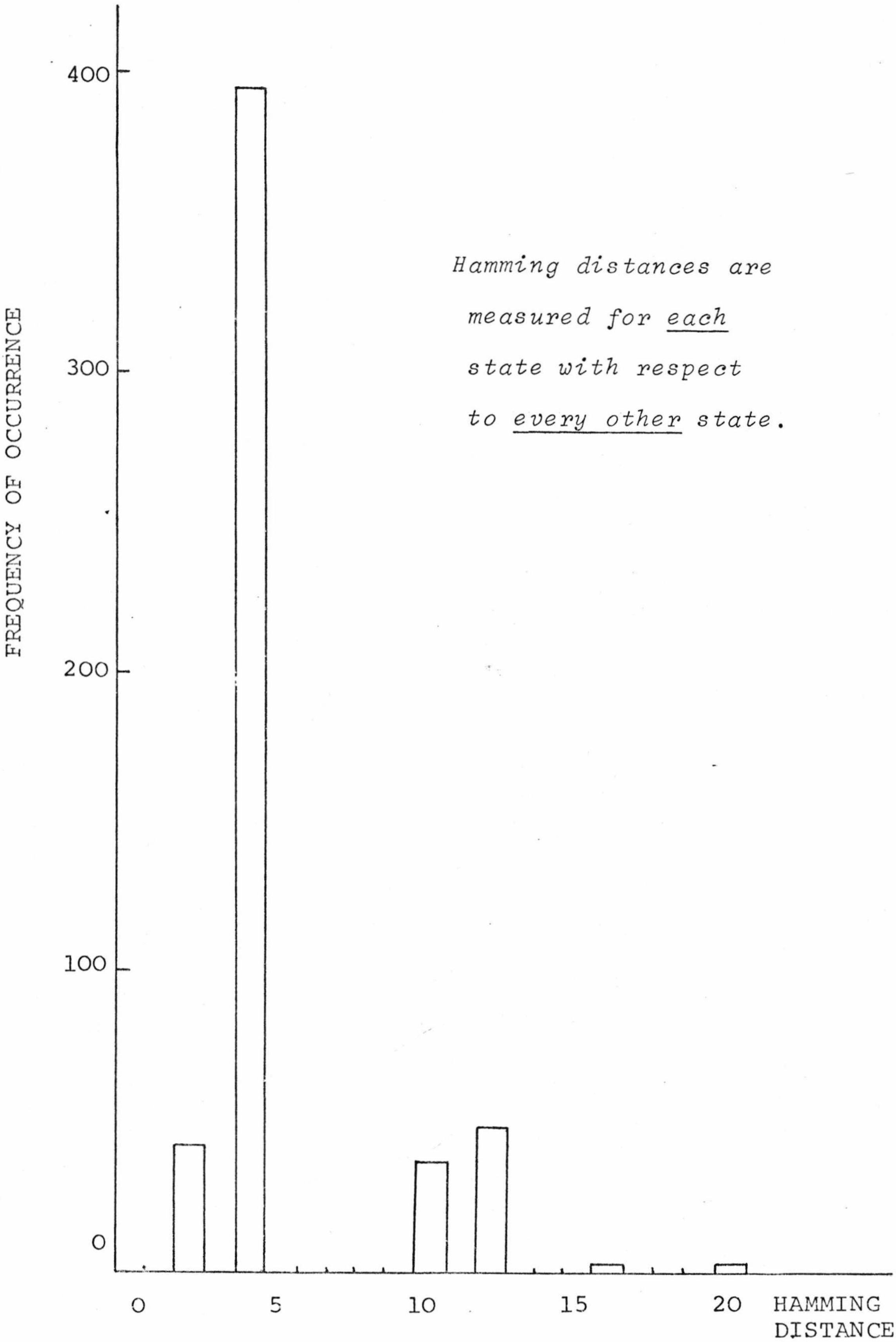


__Figure 3.6__: Hamming distance distribution for input states.

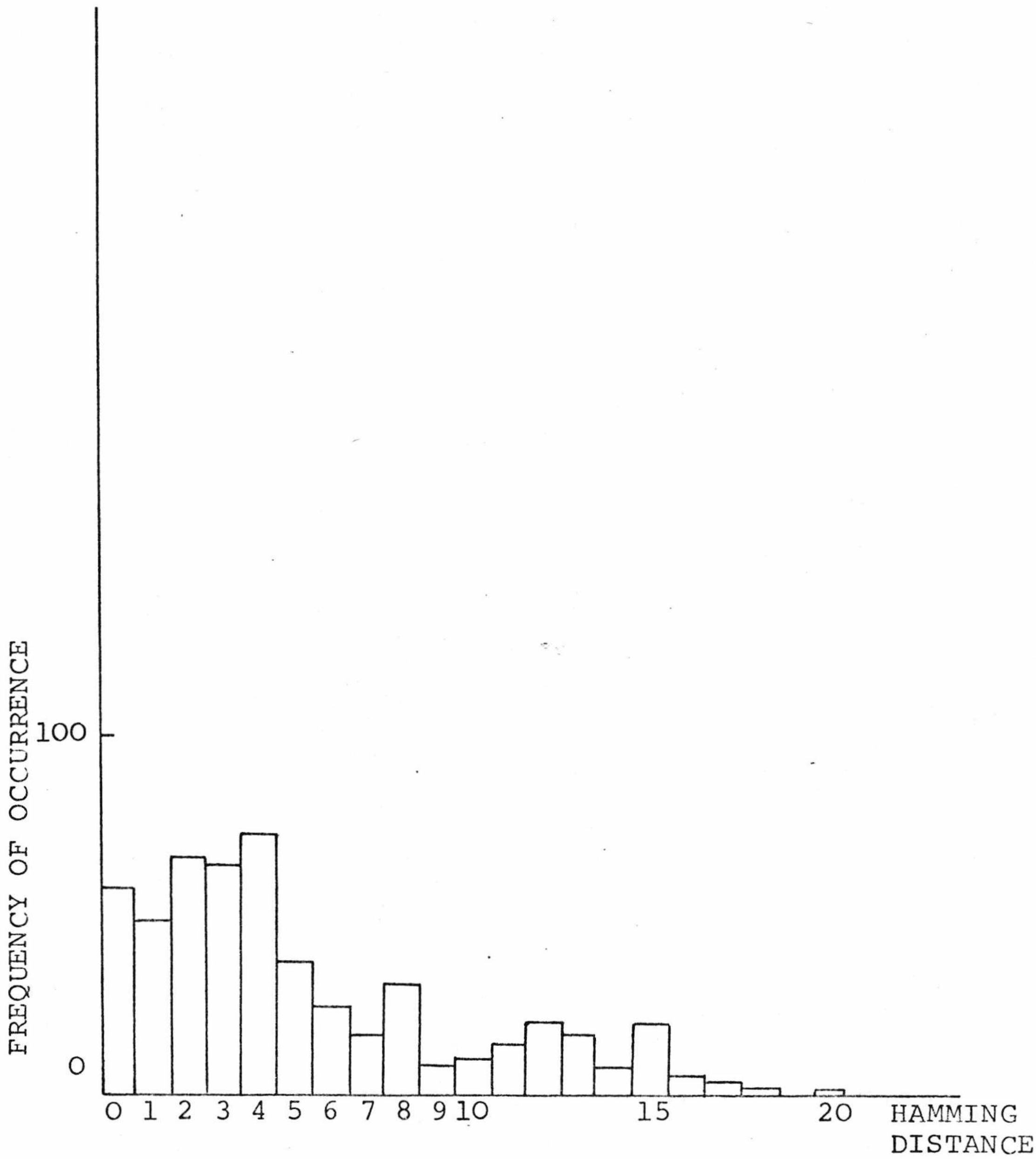
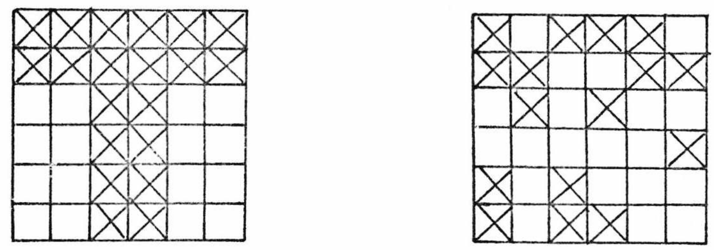
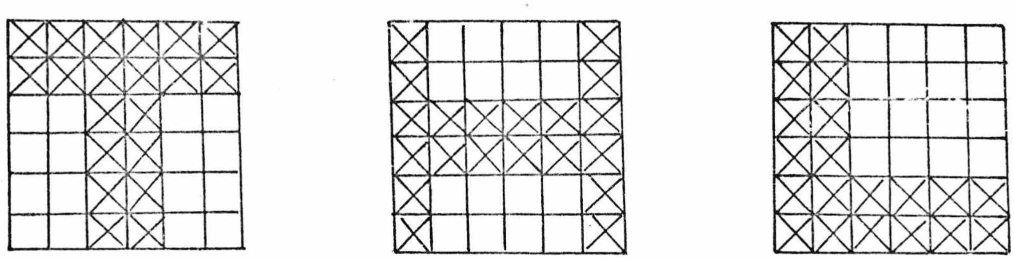


Figure 3.7: Hamming distance distribution for final states.



(a)



(b)

Figure 3.8: Generating sets for other tests.

(ii) In a case of three-class learning, using a training set consisting of three pattern classes derived from three separate generator patterns (Figure 3.8(b)).

3.5 Aging and control of the learning environment.

The results of section 3.4 have demonstrated how, during a period of learning, the net is able to formulate a model of its environment by producing clusters around internally generated archetypes of the training classes. The results presented are for a training set which was strictly controlled in the sense that individually distinct training patterns were selected from the two possible classes. It is informative to discover how the performance of the net changes under learning conditions which are not as strictly regulated.

Three tests were carried out to discover the effect of introducing random patterns (i.e. not belonging to either pattern class) at various stages during the learning period. The basic training set was the same as that used in section 3.4, with the modifications described below. The test set in this case comprised 15 typical and randomly selected T-inputs (Hamming distance =2 from the generator). For each test the final states arising in response to these test inputs were observed and the inter-state Hamming distance distributions (as described in section 3.4) measured.

The three tests were as follows:

<u>TEST 1</u> (distribution in Figure 3.9(a)) was a control experiment where no random patterns appeared during the entire training period.

<u>TEST 2</u> (Figure 3.9(b)) used the training set with 3 random patterns introduced into the sequence during the first training pass only (i.e. period of high activity at the teach terminals).

<u>TEST 3</u> (Figure 3.9(c)) used the training set with the 3 random patterns introduced into the sequence only during the third training pass with 9 teach terminals active (see Table 3.2).

Figure 3.9(a) shows the familiar distribution of clustered states where the measured Hamming distances are concentrated at the lower end of the scale with a large contribution at a Hamming distance value of 0.

The distribution in Figure 3.9(b) illustrates how the introduction of random inputs during the period of

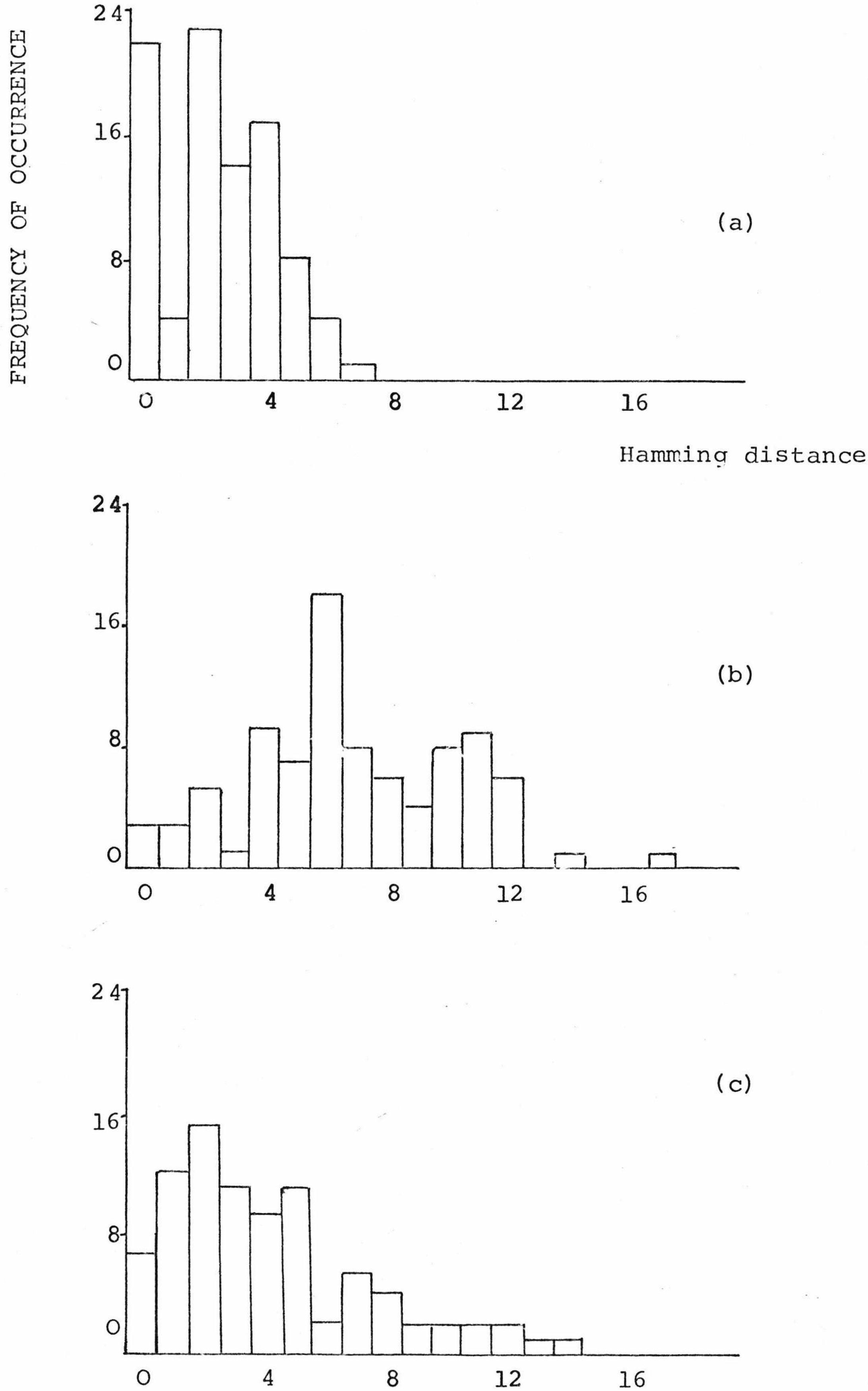


Figure 3.9: Distributions for variations in training.

high teaching activity interferes with the clustering property of the net. With respect to Figure 3.9(a) the distribution is displaced towards larger Hamming distance values - an indication that the final states produced are more dissimilar than in the previous case.

The clustering is less affected if the random patterns are only seen during the later stages of training. The distribution in Figure 3.9(c) is seen to be concentrated mainly in the low Hamming distance area and is similar in form to the distribution for the controlled training set.

We can conclude, therefore, that as is the case in human situations, it is not possible to dispense with a teacher entirely. In order to produce an efficient system, some form of control over the exposure to the environment must be imposed, but only during the period of high activity at the teach terminals.

CHAPTER 4

THE DYNAMICS OF PERCEPTION AND MEMORY

4.1 The perceptual processes of the net.

It has been established that under the maintained effect of an applied stimulus, a dynamic network is able to form associations between the pattern classes encountered during training and corresponding "recognition areas" in state space. This property is, in itself, of fundamental interest, but it must be regarded as only the first stage in the perceptual process of the net. Of equal importance is the long-term behaviour where, after initial clustering, the stimulus is subsequently removed from the net input. A study of this aspect of the behaviour of the system will permit clarification of the concepts of "perception" and "memory" in networks of digital learning elements. Some preliminary results in this field have previously been reported by Fairhurst and Aleksander (1972)[83].

4.2 Experimental method.

For this experiment, the net was connected as described previously in Chapter 3. The clustering properties of such a system have already been discussed. We shall now investigate the extent to which the recognition properties of the net are retained when the stimulus is no longer forced on the input. The following experimental procedure is adopted.

A stimulus pattern is presented to the net and held there (by virtue of the OR-ed feedback connections) until a stable state or recurring cycle of states occurs. Starting from this point the stimulus is removed (achieved in practice by forcing every element of [I] to a logical zero), and the output sequence again allowed to stabilise.

The removal of the stimulus is clearly equivalent to removing the OR gates at the net input. Since [I] is forced to zero then the input [I*] seen by the s.l.a.m.s becomes

$$\left.\begin{aligned} & i_n + o_n' \\ = \; & 0 + o_n' \\ = \; & o_n' \end{aligned}\right\} \quad \text{for all n.}$$

Consequently, under these conditions, the net is allowed to undergo unrestricted cycling on all its feedback connections.

When the output sequence again reaches a stable state or a cycle of states, the original stimulus [I] is replaced. Once more under the influence of OR-ed feedback, the state sequence is again observed until it becomes stable.

4.3 Observed results for an untrained net.

In order to appreciate the effect of training, it is first necessary to examine the system when in an untrained state.

The untrained net (stores containing randomly assigned functions) was tested with all patterns of the two groups - i.e. 630 T-inputs and 630 H-inputs (Hamming distance = 2). For each test input the experimental procedure detailed in section 4.2 was carried out.

It was found experimentally that for all the inputs tested only two different final state cycles existed when unrestricted feedback was allowed. These two cycles (referred to as Cycle A and Cycle B) were of length 16 states and 9 states respectively, and the details of the frequency with which they occurred are given in Table 4.1. As these figures show, there is no correlation between input pattern class and the final cycle entered.

CYCLE	T - INPUTS		H - INPUTS	
	Number entering cycle	% entering cycle	Number entering cycle	% entering cycle
A	544	86.3	566	89.8
B	86	13.7	64	10.2

Table 4.1: Results for untrained net.

4.4 Results for a trained network.

The net was then trained as already described in section 3.3. Once again the experimental procedure of section 4.2 was carried out to discover the residual cycling behaviour of the net in the absence of an applied stimulus pattern. The test set again consisted of all possible inputs at a Hamming distance of 2 from the two generators.

For the trained net, seven possible residual cycles were found to exist, for which the relevant data is given in Table 4.2. Here it will be noted that each cycle has a definite correspondence with one of the two trained input groups.

The classifications are seen to be:

<u>T-cycles</u>:	Cycle 1
	Cycle 3
	Cycle 5
<u>H-cycles</u>:	Cycle 2
	Cycle 4
	Cycle 6
	Cycle 7

On this basis, H-inputs are classified with zero error, while T-inputs are classified with an error of approximately 11%. The overall accuracy of the system is therefore in the region of between 5% and 6%. In effect, this second stage of perception represents a further clustering in the network. Since any one state within a cycle uniquely defines that cycle, then the net is seen to be capable of clustering 630 T-inputs into 3 states and 630 H-inputs into 4 states. The overall perceptual process of the net is shown diagrammatically

Cycle Reference	Number of States	T-INPUTS		H-INPUTS	
		Number entering cycle	% entering cycle	Number entering cycle	% entering cycle
1	12	424	67.8	-	-
2	21	38	6	183	29
3	1	132	21	-	-
4	1	33	5.2	430	68.3
5	6	2	0.31	-	-
6	18	1	0.14	2	0.31
7	6	-	-	15	2.38

Table 4.2: *Summary of experimental data for the trained net.*

for a few states in Figure 4.1.

The mechanisms underlying the behaviour of the net under the influence of an applied stimulus (i.e. with OR-ed feedback) have been discussed in Chapter 3. Now consider the case where, having reached a stable output (say S) with OR-ed feedback, the input pattern is removed. Then S, which as noted previously, will resemble the appropriate generator, will become the next input to the net, producing a response similar to S, but less so than would be the case with OR-ed feedback. The net cycles again giving rise to a new output, and so on. The likelihood of reproducing the same input pattern on successive cycles is now much less than with OR-ed feedback connections, and hence there is a tendency to produce chains of states rather than single stable states. This lengthening of cycles means that more trajectories arising from other starting states (belonging to the same class as S) are absorbed into the same cycle. The result, as the empirical data shows, is a further clustering of states into even fewer final cycles.

4.5 Comparison with single-layer net.

In order to put into perspective the results described in section 4.4, the predicted peformance of a single-layer net with no feedback was calculated using a published technique (Aleksander, 1970)[38]. The calculation was carried out for a net of 12 s.l.a.m.-8 elements

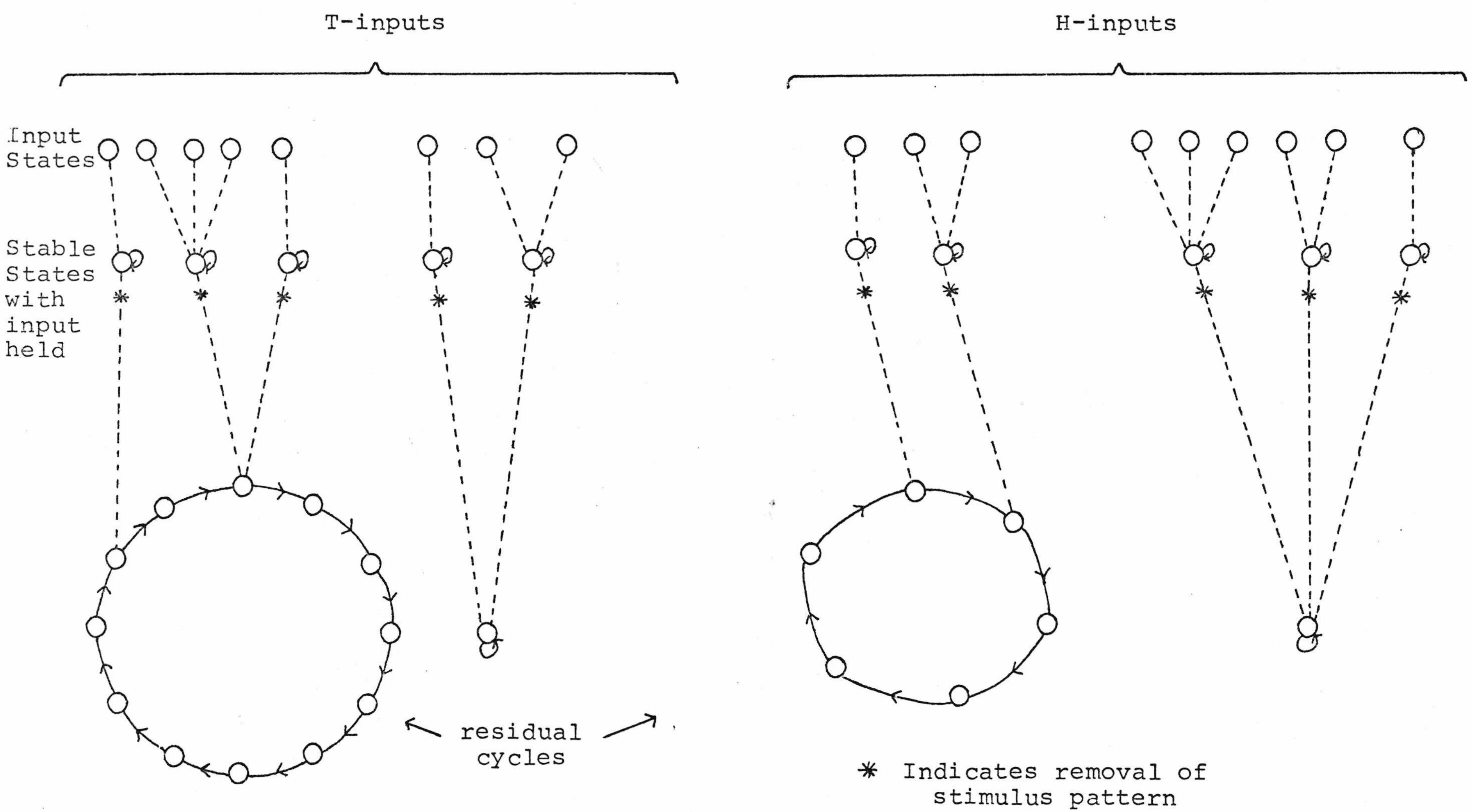


Figure 4.1: *Entry into residual cycles.*

(i.e. a complete covering of the input matrix) trained to give a logical one at every output for all the patterns of the T-group in the training set. The probable response to a test T-input and a test H-input was calculated by assuming average Hamming distance values between test patterns and training patterns.

The predicted behaviour of the net is described by the histograms in Figure 4.2. Here it is seen that for typical inputs, a very simple network can be used to separate the two pattern classes, merely by the introduction of suitable circuitry to set a threshold level between the response to Ts and the response to Hs. There are, however, two factors which illustrate the effectiveness of the dynamic system. In the first place the distributions shown in Figure 4.2 are calculated for "average" test inputs. If the precise Hamming distance values for a selection of inputs are used in the calculations then the predicted separation of response distributions will, in some cases, be considerably less well-defined.

The second factor is even more significant. Table 4.3 shows the output vectors, [O] - represented as a 36-bit word in each case - for <u>all</u> states in <u>all</u> the residual cycles classified as T-cycles, while Table 4.4 similarly shows all states in all the H-cycles. If the two are compared bit by bit it will be seen that bit 5 is set to logical 1 for <u>every</u> T-state, and to logical 0 for <u>every</u> H-state. This points to a very powerful discriminating factor for classification of the two pattern groups, since

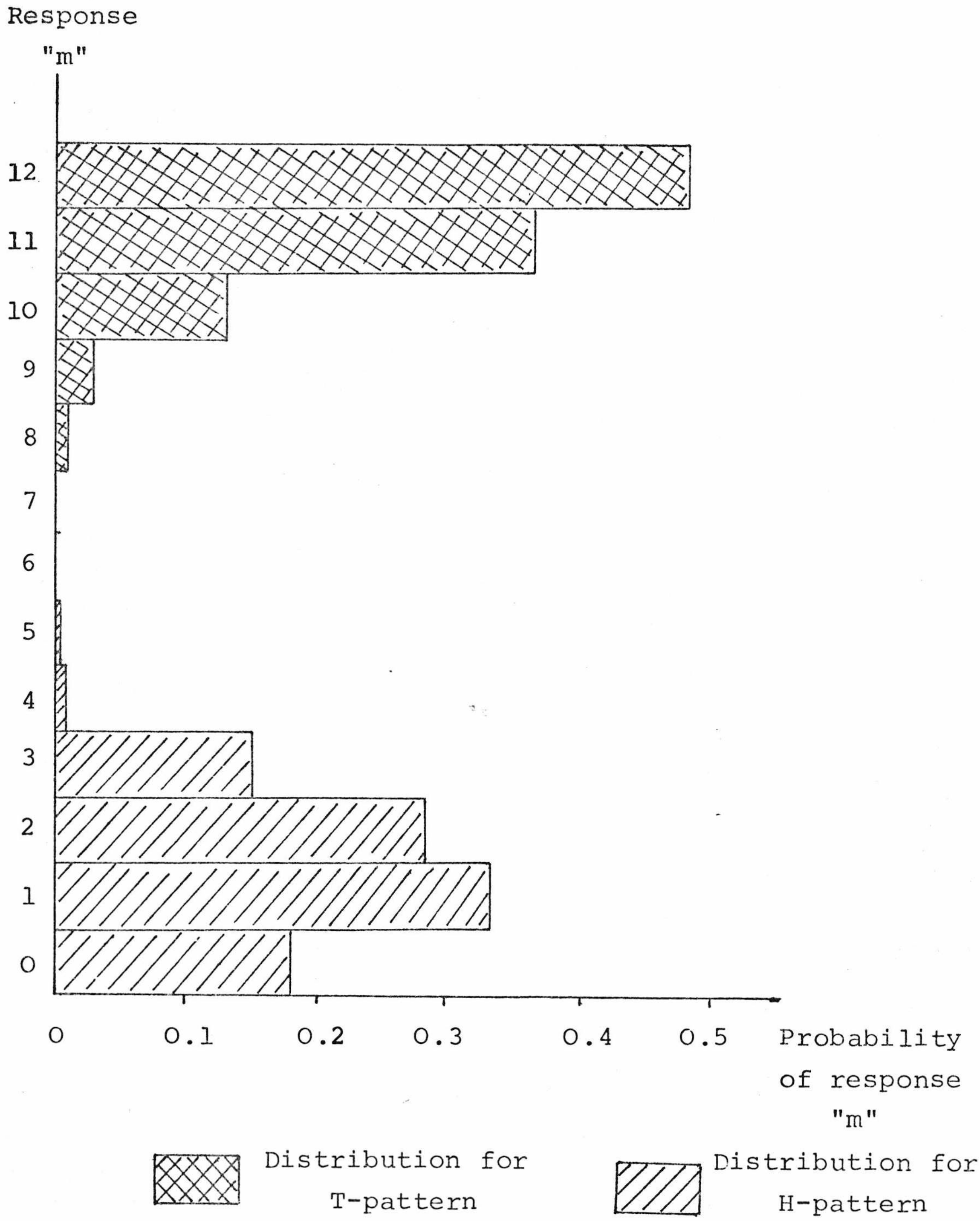


Figure 4.2: *Predicted response of static network.*

Bit 1, Bit 5, ..., Bit 36

Output vector	T-cycle
111011100101011101000101001000000000	Cycle 1
101111100101011001110000001101100000	
101011101101001101010001001011000000	
001011100101011101000011000001000000	
101011100001011001010001001101110100	
101111100111011001010000001001000000	
101011110101001001010001001101001000	
101011100101011101000001001001000010	
101011100111011001010001001101000001	
111011110101011001010101001101001000	
101111100101011101010000001101100010	
101011100101001101010001001001000001	
101011100101011101110001001101000000	Cycle 3
011011110101001001000111000001001000	Cycle 5
101110100001011001100000001001100010	
101111101101001001010000001011000001	
011011100101001100000111000000000000	
111110100001011001101000001001100110	
101110101111001101010001001011000001	

Table 4.3: Output vectors in T-cycles.

Bit 1 Bit 5 — — — — — — — — — — Bit 36

10110010000100111111000000000101100000	Cycle 2
101101111001001011000010000011110000	
001101100001001010000000000110011 0000	
111101100101001001001000000101010110	
101000100011001111000001000001100001	
111101110000011111000100000000111 1000	
101000100001001011110000000001100010	
101101111001011111000010000011110001	
011101100001001010010100101000110000	
111100100101001101101000000101100110	
101C00111011001111000001000011100001	
111101110001011110000100001001111000	
101000100101001011111000000001100000	
101001111001011111000011000011110000	
001101100001011010010000001001110000	
101101100101001101011000000001010110	
101001100011001011000001000001100001	
111101110001011111000100000101111000	
101000100001001011010000000001100010	
101101110001011111000000000001110001	
111101100001001011010100000001110000	
10110010000100111111000000000101100000	Cycle 4
101101100001001011000000000101110000	Cycle 6
111101100101001011011100000101110010	
101100100001001111110000100010110000 1	
111101111001001011000110000011110000	
001101100001001010100000000110010000 0	
111101101101000001001010000101010010	
101001100000101010000000100000110000 1	
111101100101001011011100100101110010	
100100100001101111111000100000110000 1	
111101111001001111001110100001110000	
100001100001101010100001000100100000	
111101101001010011001010000010111001	
101101100000101010010001000000110000 1	
111101100101001111011100100101110010	
100100100001101111110000100010110000 1	
111101111001001011001110000000111 0000	
111101100101011111011100000101110010	Cycle 7
101100100001001111111000110000110000 1	
111101111001001111000110000001111 0000	
001001100001001010100000000110010000 0	
111101101101010001001010000101010010	
101001100000101010010001000101100001	

<u>*Table 4.4:*</u> *Output vectors in H-cycles.*

in the case of the dynamic net it is only necessary to detect the presence or absence of a single bit in the final state cycle in order to separate the two sets of inputs.

Further examination of individual states shows that no other single bit provides <u>unique</u> identification of a cycle, but a high degree of accuracy can be achieved in other cases as follows. For convenience, only the four main cycles - Cycle 1 and Cycle 3 for T-inputs, and Cycle 2 and Cycle 4 for H-inputs - will be considered.

Examination of bit 17 shows it to have a value of 0 for Cycles 1 and 3 (T) and a value of 1 for Cycle 4 (H). This immediately allows one-bit separation of 556 T-inputs and 430 H-inputs. In addition, bit 17 is a logical 1 for 18 out of a possible 21 states in Cycle 2 (H-cycle). Assuming an equal probability of occurrence for all states in this cycle, the most probable number of inputs entering Cycle 2 which can be correctly identified by observing only bit 17 will be

$$\frac{18}{21} \times \underset{\substack{\uparrow \\ \text{(total number of inputs} \\ \text{entering Cycle 2)}}}{183}$$

$$\approx \; 157$$

Thus, considering the correct classifications in terms of the total number of inputs entering these cycles, calculations show figures of 100% recognition of T-inputs and about 96% recognition of H-inputs by observing only bit 17. The same argument can be applied to other one-bit classifications, and some examples are given in Table 4.5.

Bit Number	Correct Classification of T - inputs (%)	Correct Classification of H - inputs (%)
4	81	90
17	100	96
27	93	94
31	81	97

Table 4.5: One-bit classification

Returning to the single-layer net comparison, it is possible to predict the probability of <u>one-bit separation</u> of the two classes in this case. Suppose a net of k n-input elements samples an input retina of of N points. Then, following the early part of the analysis given in Reference 38. . . .

For a test pattern which is at Hamming distance H_j from the j^{th} pattern of a training set of S patterns ($1 \leqslant j \leqslant S$), then the probability of any one particular s.l.a.m. producing a 1 at its output is given by

$$P_j = \frac{\binom{N-n}{H_j}}{\binom{N}{H_j}}$$

and therefore the probability that a 0 is generated is

$$Q_j = 1 - P_j .$$

Considering the complete set of S training patterns, the probability that a 0 is generated is given by

$$Q_r = \prod_{j = 1,S} Q_j \qquad \ldots \quad \ldots \qquad (4.1)$$

and the probability that a 1 is generated is

$$P_r = 1 - \prod_{j = 1,S} Q_j \qquad \ldots \quad \ldots \qquad (4.2)$$

Consider the single-layer net previously described, which is trained to generate 1s at each element output for the T-patterns of the training set. For test inputs T_1 and H_1 (a T-pattern and H-pattern respectively), the probability that any particular element will generate a 1 in response to T_1

$= P_{T_1}$ (calculated from Equation 4.2 substituting Hamming distance values of T_1 from the patterns of the training set).

Also, the probability that an element will generate a 0 in response to H_1

$= P_{H_1}$ (using Equation 4.1).

$\therefore$ The probability that any particular element will generate a 1 for T_1 <u>and</u> a 0 for H_1 is given by

$$P_{TH} = P_{T_1} \times P_{H_1} \qquad \ldots \quad \ldots \quad \ldots \quad (4.3)$$

Extending this principle to the case where t test patterns of the T-group and h test patterns of the H-group are considered, the probability of separation by observing only one bit is given by

$$P = \prod_{i=1,t} P_{T_i} \times \prod_{j=1,h} P_{H_j} \quad .. \quad .. \quad (4.5)$$

Substituting values for the single-layer net as before, a predicted probability of 0.745 is obtained for one test pattern of each group. By the time 20 test patterns of each group have been considered the probability of one-bit separation has decreased to 0.0026. In other words, for the large set of test patterns used in the experiment described for the dynamic net, the probability of one-bit separation is negligible.

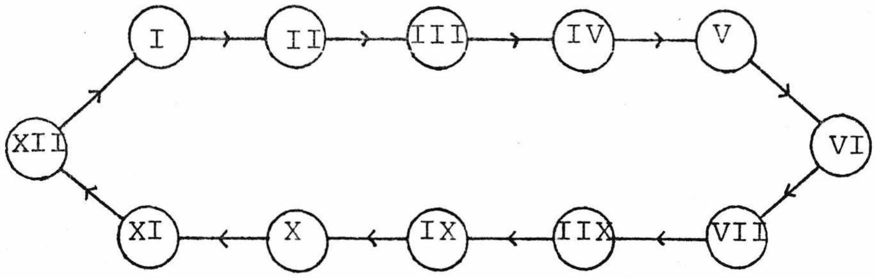
4.6 Switching between the two learnt classes.

We have seen how the system enters one of few residual cycles when allowed unrestricted feedback in the absence of a forced stimulus. It is also interesting to observe the behaviour of the system when an input is subsequently forced back on to the net while in one of these residual cycles.

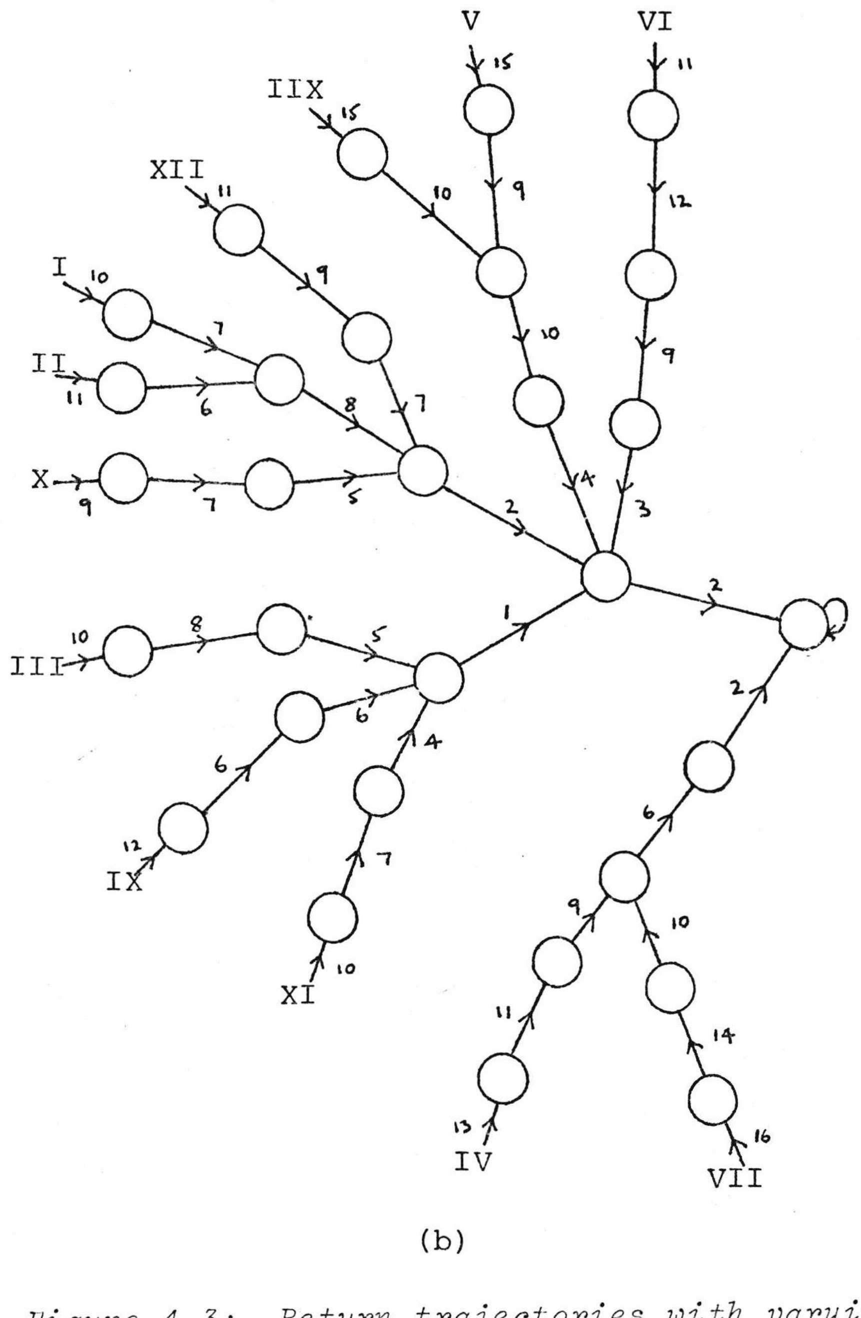
As an example, consider the case where in response to a T-input, a T-cycle is eventually entered after the original stimulus is removed. Replacing the same stimulus is found to drive the system back into the stable state previously reached with the stimulus held at the input. This behaviour is found to be independent of the exit point from the residual cycle (i.e. the point at which the stimulus is replaced).

For example, consider a typical T-cycle as shown in Figure 4.3(a), where the states are identified by Roman numerals. The return trajectories on replacing the stimulus at each possible exit point are shown in Figure 4.3(b). Roman numerals again denote exit point from the cycle, and the Arabic numbers refer to the transition Hamming distance between successive states. The form of the state diagram is a familiar tree-structure, with decreasing transition Hamming distances as stability is approached. The final state (T) is identical with the state first reached with OR-ed feedback in response to the same stimulus. This is another illustration of the way in which OR-ed feedback connections allow the applied stimulus pattern to control the subsequent state trajectories, as discussed in Chapter 3.

More general behaviour is illustrated by Figure 4.4, which shows the overall dynamics of the system for a particular set of cycles. A T-input is applied to the net and held there until a stable state is reached as before. If this stimulus is removed the net drifts into a T-cycle consisting, in this case, of a recurring sequence of 12 states (Figure 4.4(a)). Replacing the T-input returns the system to the previously entered stable state (Figure 4.4(b)). If,however, an <u>H-input</u> is forced on to the net while in the residual T-cycle (Figure 4.4(c)), the state trajectory follows a new path which ends in a recurring cycle of 3 states, these being close in Hamming distance



(a)



(b)

<u>*Figure 4.3:*</u> *Return trajectories with varying cycle exit points.*

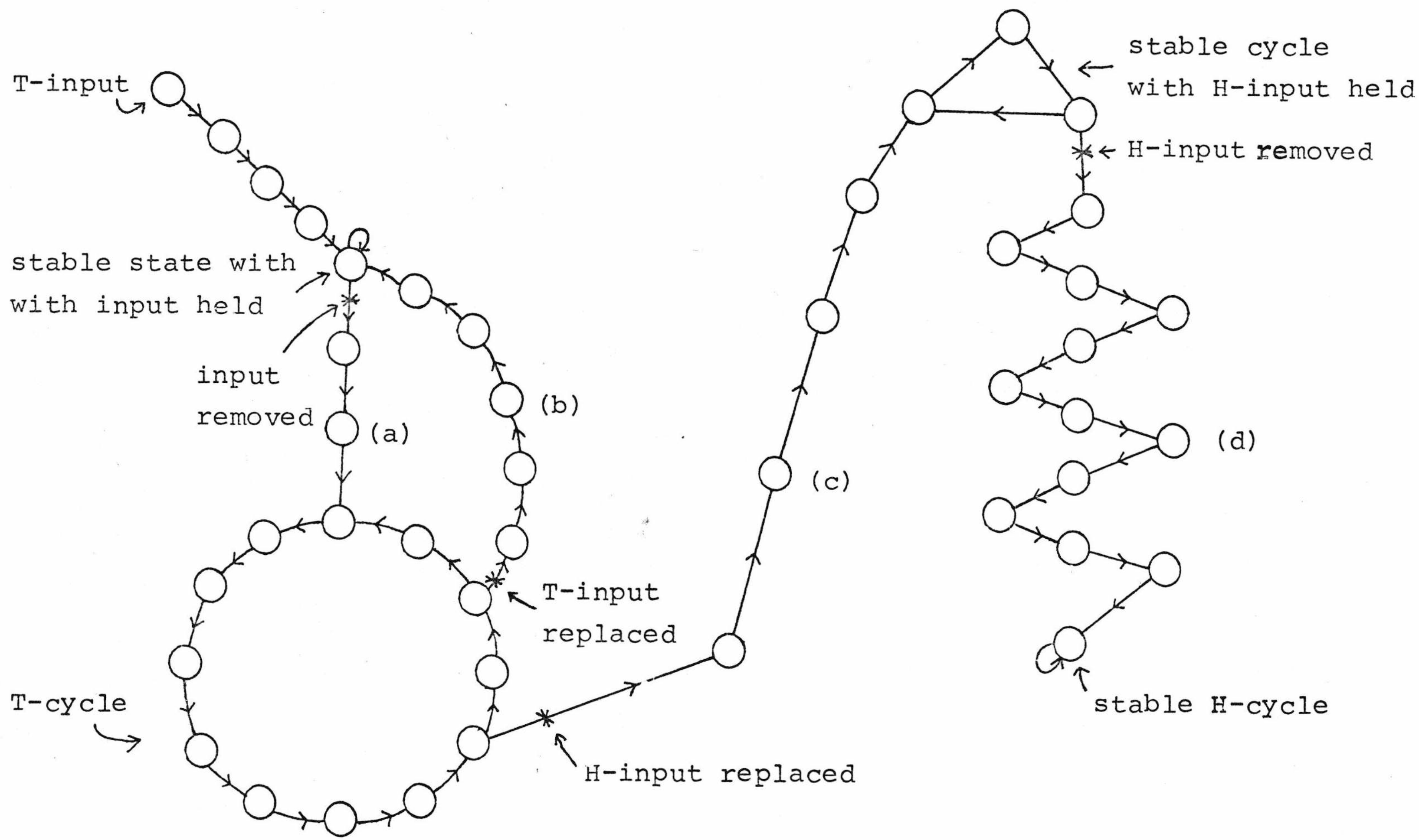


Figure 4.4: *Overall state space dynamics.*

to the H-generator. Finally, when this H-input is removed (Figure 4.4(d)), the system does not fall back into the T-cycle from which the sequence orginated, but moves into a familiar residual cycle associated with patterns of the H-group.

This behaviour again demonstrates the powerful controlling effect of the OR function which was discussed in Chapter 3. It is clear that the net is able to retain its recognition ability even in the absence of an input stimulus pattern. The controlling effect of the pattern at the input "switches" the net into areas of state space associated with that input during training.

4.7 A comment on comparisons with human memory.

It is now generally accepted, on the grounds of both introspective and experimental evidence (Slamecka, 1967)[84], that two different types of memory are found in humans. One type is of a transient nature and stores information of immediate events (short-term or primary memory), while the second contains more permanent memory traces (long-term or secondary memory).

This phenomenon has been studied both by psychologists and neurophysiologists, and it is of importance that, despite having distinct properties, the two types of memory need not be physically different, but merely different aspects of the same structure. Also, psychological theories (Norman, 1968)[85] propose that sensory information first enters the short-term memory and can subsequently, by various processes (repeated presentations, rehearsal etc. (Waugh & Norman, 1965)[86]) be transferred to the long-term memory store. In physiological terms (Deutsch, 1967)[87], short-term memory is seen as consisting of "reverberations" in the electrical activity of groups of neurons, while long-term memory occurs as the result of more permanent physical changes in the neuron assembly (e.g. strengthening of synaptic connections).

Similarly, the existence of two types of memory is evident in learning networks, and these too can be considered as two different mechanisms within one single structure. Here it is possible to distinguish between a long-term memory effect (the residual cycling of the net) which stems directly from the physical structure of the system (i.e. the actual logic functions contained by the elements themselves) and a short-term memory effect - caused by the OR-ed feedback connections - which is dependent on immediate external events (i.e. the particular stimulus which is present at the time).

4.8 Overall state space structure.

The previous experiments have considered only those inputs and corresponding states located in very confined regions of the state space in the immediate neighbourhood of the generator patterns. A broader picture of the overall state space structure of the system can be obtained by considering test inputs and states located at points more distant from the generators in the state diagram.

To test the net exhaustively with all possible inputs is clearly impracticable, since the number of possibilities becomes prohibitively large (on a 6x6 matrix there are 2^{36} possible inputs). Even to choose a complete set of patterns at selected Hamming distances becomes progressively more difficult as Hamming distance increases. For instance, there are

$$\frac{36!}{26! \times 10!}$$

$$\simeq 3 \times 10^{10}$$

possible patterns at a Hamming distance of 10 from a generator.

A sampling approach is therefore suggested in order to keep within the limits of feasibility. A sample of 10 patterns was chosen for Hamming distance values of 2, 4, 6, ---- 26 with respect to each generator. This gave a complete test set of 260 inputs. For each input the same experimental procedure described in section 4.2 was carried out in order to estimate the total number of possible residual cycles of the net.

For the set of inputs tested, only the 7 cycles previously encountered were found, and no additional cycles occurred. On the basis of the previous classification, the entire state space of the system can therefore be divided between T-regions and H-regions. Figure 4.5 shows the relation between the location in state space of the test inputs and the residual cycle classification. It can be seen that the probability of entering a T-cycle is greatest for inputs in the regions of the T-generator, and conversely, the probability of entering an H-cycle is greatest for inputs around the H-generator. There is no distinct dividing line in state space, but rather a steady change in probabilities as a function of distance from the generator.

The results obtained here can be used to sketch these probability functions, as illustrated in Figure 4.6. It is seen that the distributions for both T-inputs and H-inputs show a definite downward trend as the Hamming distance from the appropriate generator increases. It is also interesting to note that the probability function for T-inputs has a different shape from that for H-inputs. A possible explanation for this is the fact that far more states were involved in H-cycles than in T-cylces. The H-cycles are thus more likely to "sweep up" state trajectories with the result that the probability of entering an H-cycle decreases less sharply than in the case of the T-cycles.

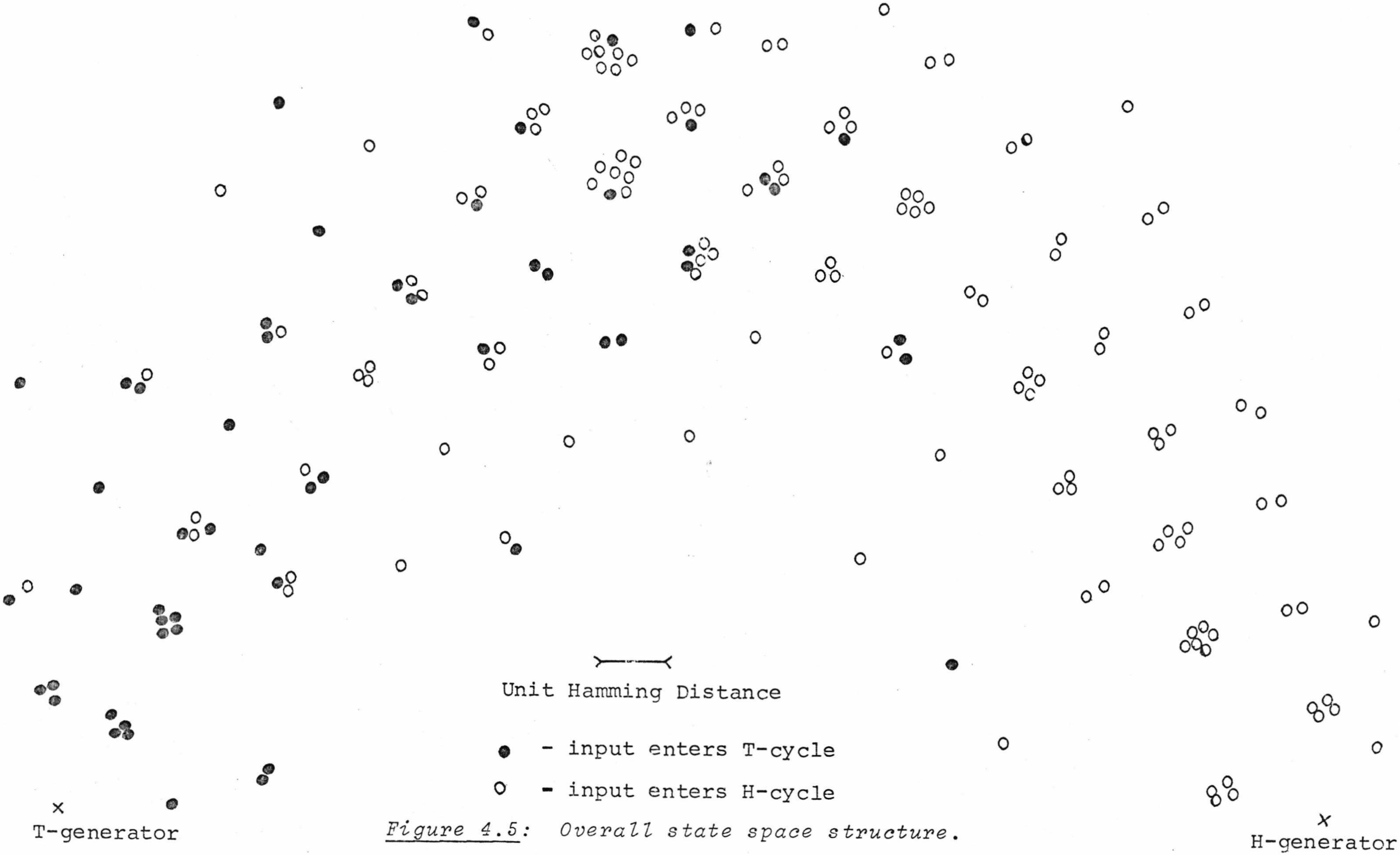


Figure 4.5: *Overall state space structure.*

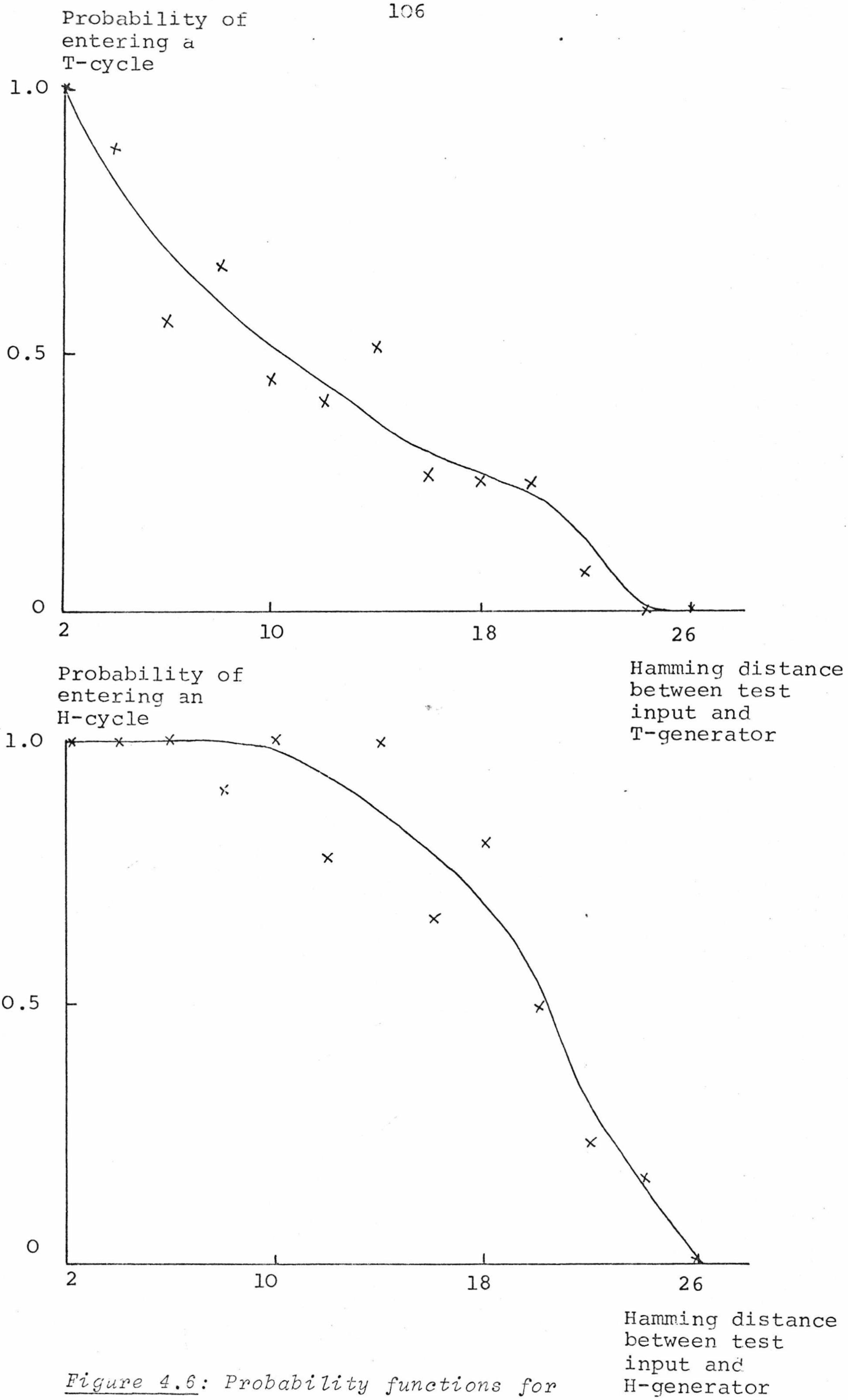


Figure 4.6: Probability functions for T- and H-cycles.

Two main points can be made in conclusion. First, "perception" by the system can be defined as the entry into specific areas of state space which are determined by the classes of input seen by the net during training. Second, the existence of few and distinct cycles related to classes of input demonstrates that a memory of the trained classes is retained in the absence of a direct stimulus, and shows how the dynamic net is forced to "live" in state space regions related to its training environment.

CHAPTER 5

RECOGNITION OF PATTERN SEQUENCES

5.1 The use of contextual information.

In the preceding chapters where simple pattern recognition tasks have been described, the discussion has been confined to situations where a network processes a single input frame, producing a response which enables the input to be assigned to one of a number of possible classes. Up to this point, no mention has been made of the case where sequences of inputs, which are meaningful by virtue of their spatial or temporal relationships, are to be classified. Since in practical situations patterns are rarely seen in isolation, the application of digital networks to the processing of pattern sequences could be of considerable importance.

The utilisation of context is obviously of supreme importance in human processing of perceptual data. In particular, language - which by its very nature is highly redundant - provides a typical illustration of the ability of humans to make use of context. Because of the syntactic and semantic restrictions on the possible sequences of letters and words which make up sentences, the average reader or listener is able to understand written or spoken language without specific processing of each individual character or even each individual word.

Empirical evidence for increased efficiency as a result of contextual processing in human subjects has been clearly presented by Miller (1962)[88]. In his experiments on the perception of speech he shows that performance on a list of words where each successive word is one of a limited number of alternatives is almost identical to performance on lists of words which form "grammatical" sentences. Conversely, "pseudo-sentences" (e.g. sentences in reverse order) gave results similar to those observed when successive words in a list were drawn from a much wider vocabulary. In other words, the results show that humans use context to reduce the number of alternatives which can occur in successive steps in a sequence of perceptual events.

This aspect of human information processing has been incorporated into several psychological models such as that proposed by Morton (1969)[89], and similarly in problems of pattern recognition by machine, context has not been overlooked as a means of improving efficiency. The theoretical background to contextual analysis has been laid down by Abend (1966)[90], and practical applications considered by various authors who report considerable reduction in error rates as a result of using contextual information. (Duda and Hart, 1968[91], Vossler and Branston 1964[92]).

The purpose of this chapter is to present briefly some results which demonstrate that a net of s.l.a.m. elements can learn to respond to pattern <u>sequences</u> as distinct from individual frames. The evidence shows that digital learning networks can make use of context by being sensitive to input sequences.

5.2 Training procedure.

The block diagram of the system is unchanged, remaining as was shown in Figure 3.1. In order to train the net so as to make it sensitive to a sequence of patterns, it is only necessary to introduce each frame of the sequence so as to coincide with the arrival of successive clock pulses and assume a unit delay in the feedback path. The teach terminals again sample the input and an association is formed between each frame and the OR-ed amalgamation with the previous frame. Thus, to train a series of n frames (F_j) in the sequence

$$\begin{tikzcd} F_1 \arrow[r] & F_2 \arrow[r] & F_3 \arrow[r] & ---- \arrow[r] & F_n \arrow[llll, dashed, bend left] \end{tikzcd}$$

training proceeds according to Table 5.1.

5.3 Pattern sequences for experimentation.

In the previous experiments on recognition of single frames, the test patterns were produced by defining a generator (prototype) pattern and randomly inverting a

Clock Pulse	Input to s.l.a.m.s	Trained Response
1	F_1	F_1
2	$F_1 + F_2$	F_2
3	$F_2 + F_3$	F_3
⋮	⋮	⋮
n	$F_{n-1} + F_n$	F_n
n + 1	$F_n + F_1$	F_1

(+ signifies logical OR)

Table 5.1: *Training steps.*

number of elements of the digitised pattern to give a desired Hamming distance value.

In the same way, it is possible to define a "prototype" or generating <u>sequence</u> (Figure 5.1), which in this case consists of three frames (referred to as $A \rightarrow B \rightarrow C$) in the form of a vertical bar which moves from left to right across the input grid in three steps. Any frame in the sequence can be degraded to the desired Hamming distance as previously described.

5.4 Response to distorted sequences.

Using the scheme previously described (see section 3.4) the net was trained on the prototype sequence $A \rightarrow B \rightarrow C$. On completion of training the net was tested on 15 sequences ($A \rightarrow B_1 \rightarrow C$, $A \rightarrow B_2 \rightarrow C$, -- $A \rightarrow B_{15} \rightarrow C$) where in each case the second frame (B_n) was degraded by a Hamming distance of 2 with respect to the original frame B. For each test the sequence of outputs was observed, and Hamming distances measured with respect to the prototype patterns A, B, and C.

The first point of interest noted about the results was that in response to slightly degraded versions of the trained sequence, the net produces series of patterns which retain the properties of a moving bar, culminating finally in a recurring cycle of three states corresponding to the three frames A, B, and C. A typical state trajectory

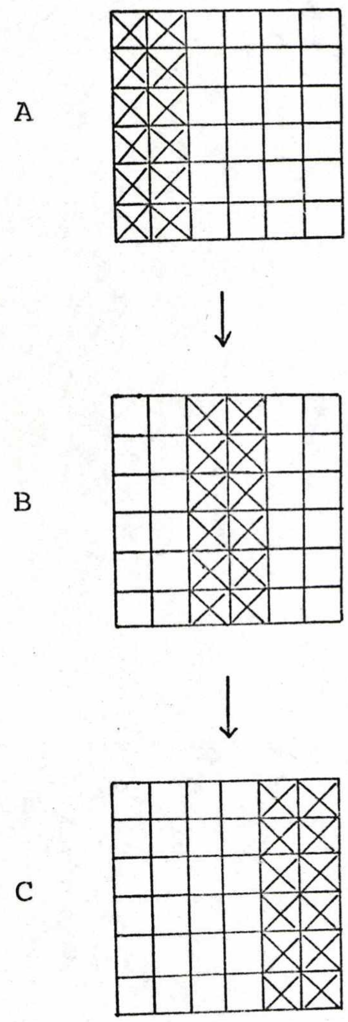


Figure 5.1: Prototype sequence.

(consisting of 12 states) is shown in Figure 5.2.

For all 15 tests the final output cycle is found to maintain synchronism with the applied sequence of stimuli. In other words, in response to a slightly degraded input sequence, the net produces a sympathetic generation of a similar sequence at the output. The definition of "recognition" of a sequence in these terms was first introduced by Aleksander (1970)[73].

If the final response cycles are examined more closely they are found to point to an interesting parallel with the clustering effect noticed for single frames (see section 3.4). For the 15 input sequences, only 10 different final sequences are found, one of these occurring 4 times. This most frequently occurring sequence (Figure 5.3) is analogous to the "archetype" state generated by the system in single pattern processing.

The results can be presented graphically by representing each frame of the most frequently occurring sequence (the "archetype" sequence) as the vertex of a triangle, indicating the relation of other sequences by measuring Hamming distances of corresponding frames from these reference points. Figure 5.4 illustrates the principle, and Figure 5.5 shows some typical results plotted in this way.

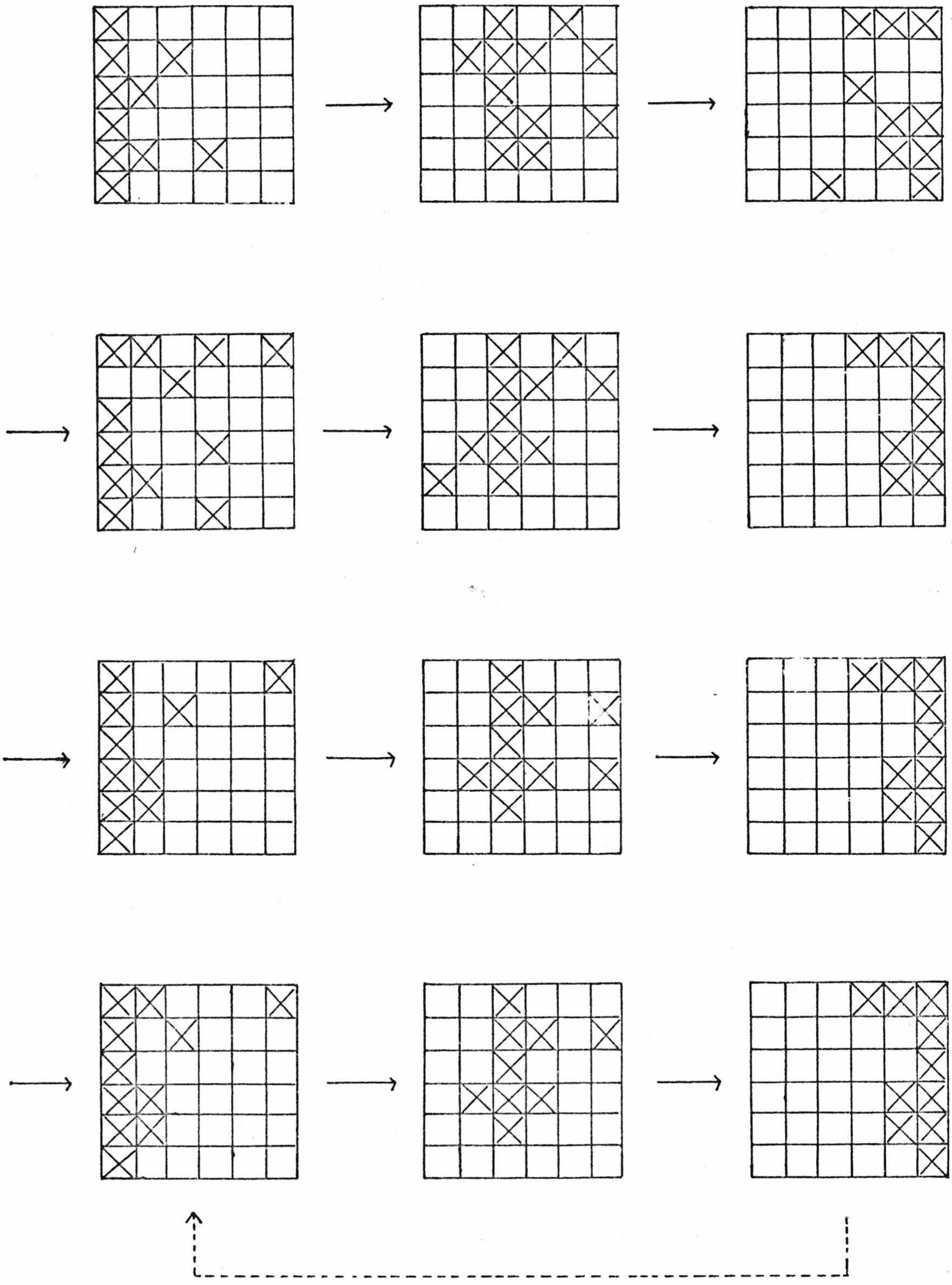
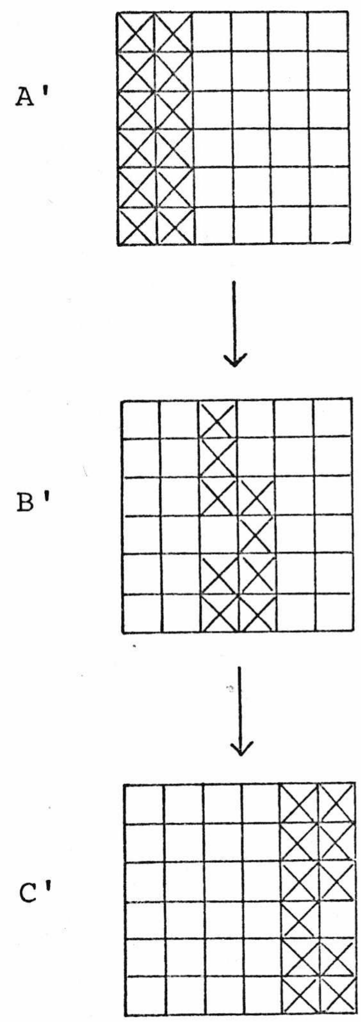
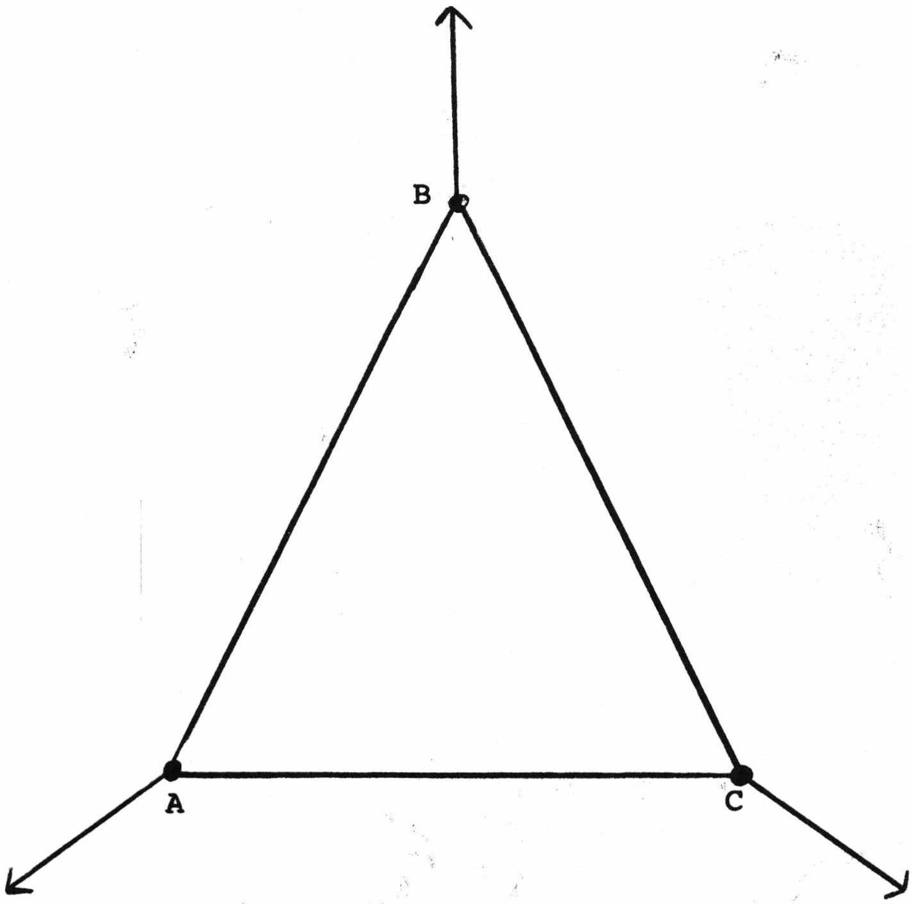


Figure 5.2: *Typical output sequence.*



<u>Figure 5.3</u>: Most frequently occurring final state sequence.

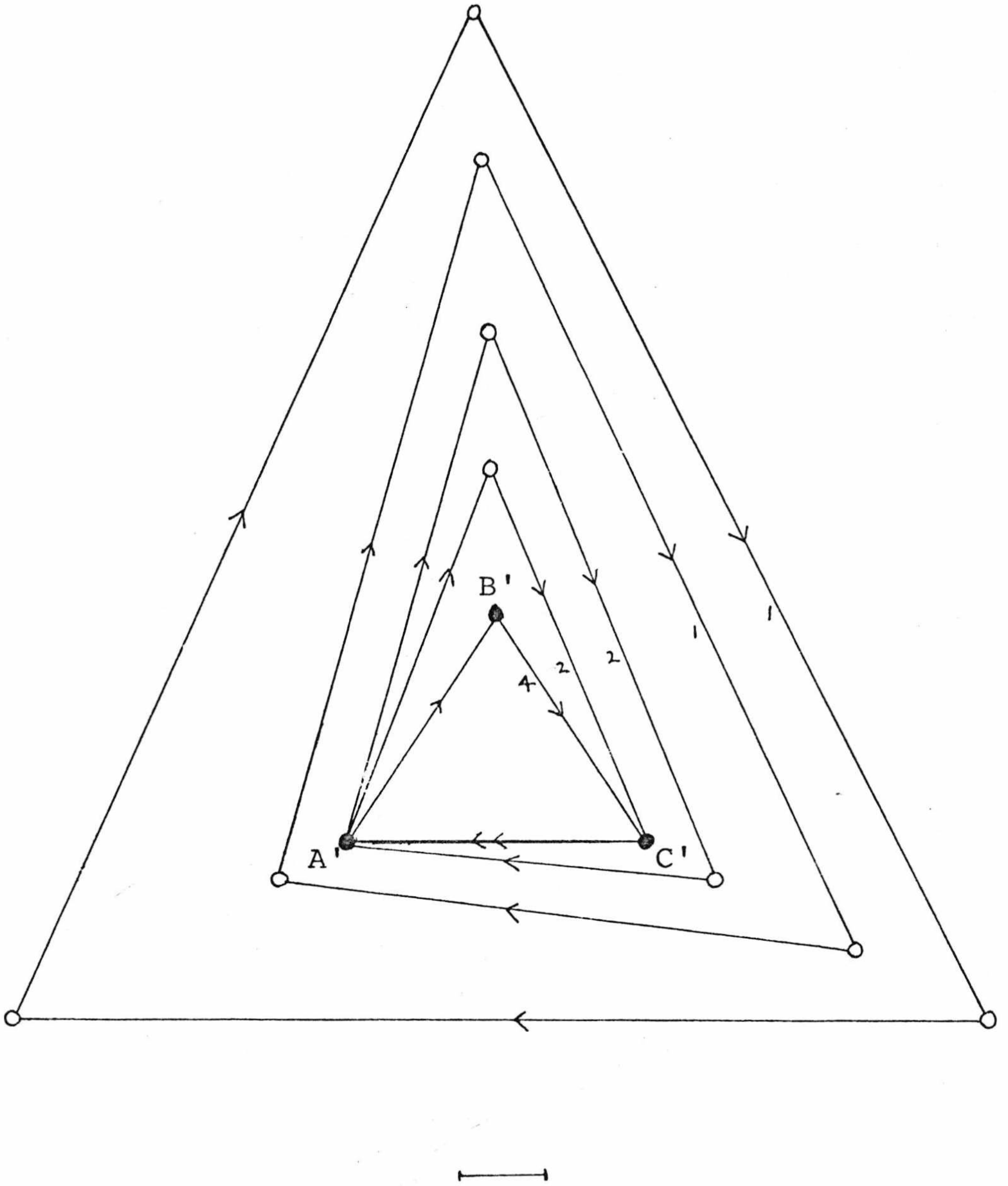


Direction of increasing Hamming distance of any frame A_n with respect to A.

$A \rightarrow B \rightarrow C$ refers to the "archetype" sequence.

Figure 5.4: Method for presenting results

As a comparison, the sequences were also presented to an untrained net with logic functions randomly assigned. The output sequences consisted of a number of meaningless patterns which bore no relation to the moving bar at the input. Eventually, recurring cycles of states (typically of length 6 states) were entered.



● - A', B', C' represent the archetype sequence

The small figure refers to the number of times each cycle is entered out of 15.

Figure 5.5: Typical results.

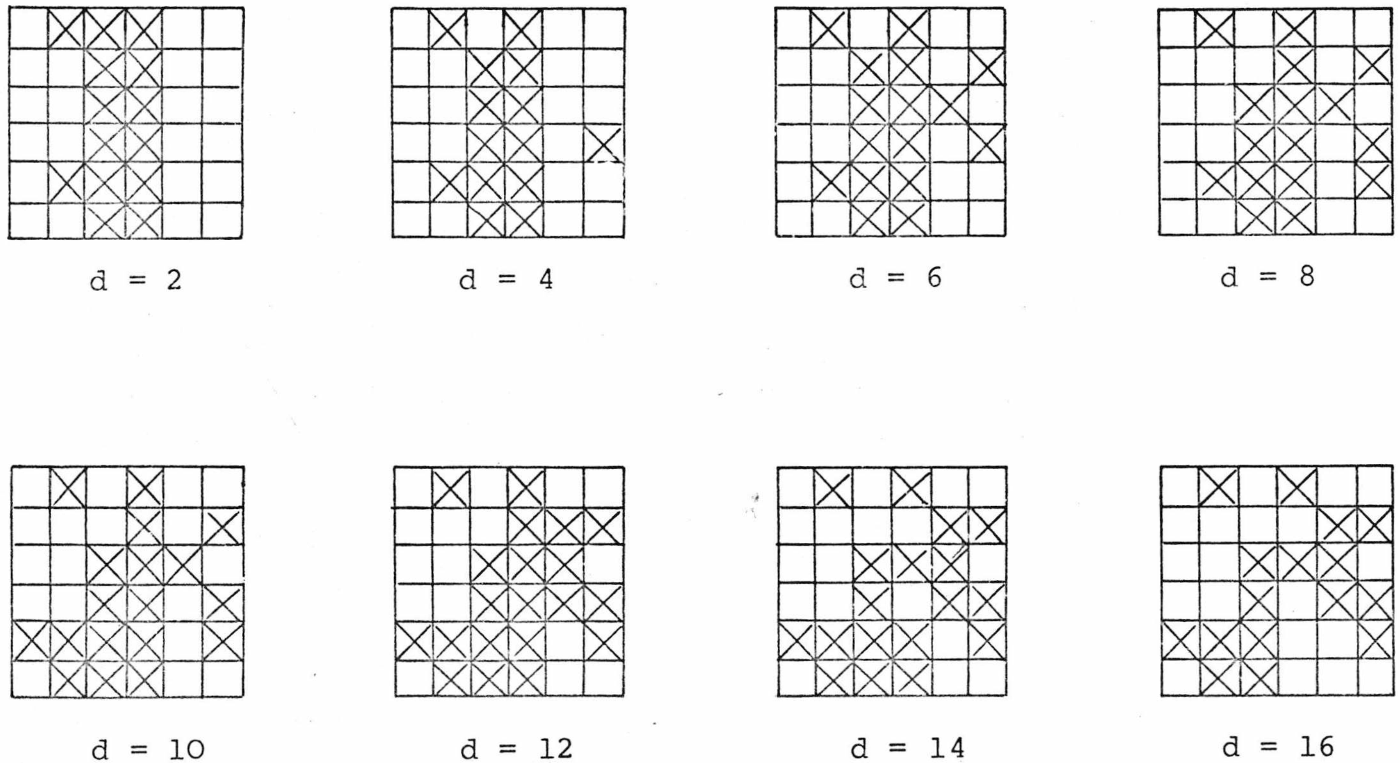
5.5 Increasing distortion of sequences.

The net was again trained on the generating sequence. Subsequently, the trained net was tested with the sequence ten times, the second frame of the sequence (B_n) becoming progressively more distorted (the Hamming distance between B and B_n increasing) with each test, as Figure 5.6 shows.

The resulting state space behaviour follows the general form shown in Figure 5.7. It can be seen that the characteristics of the moving bar are largely retained (e.g. see the cycle produced when the middle frame is distorted by a Hamming distance of 12, as shown in Figure 5.8), and synchronism between input stimulus and the output cycle is maintained. This type of behaviour holds until, at the point where the distortion of frame B has reached a Hamming distance value of 16 compared with its generator, synchronism breaks down and a recurring cycle of 6 states (Figure 5.9) occurs.

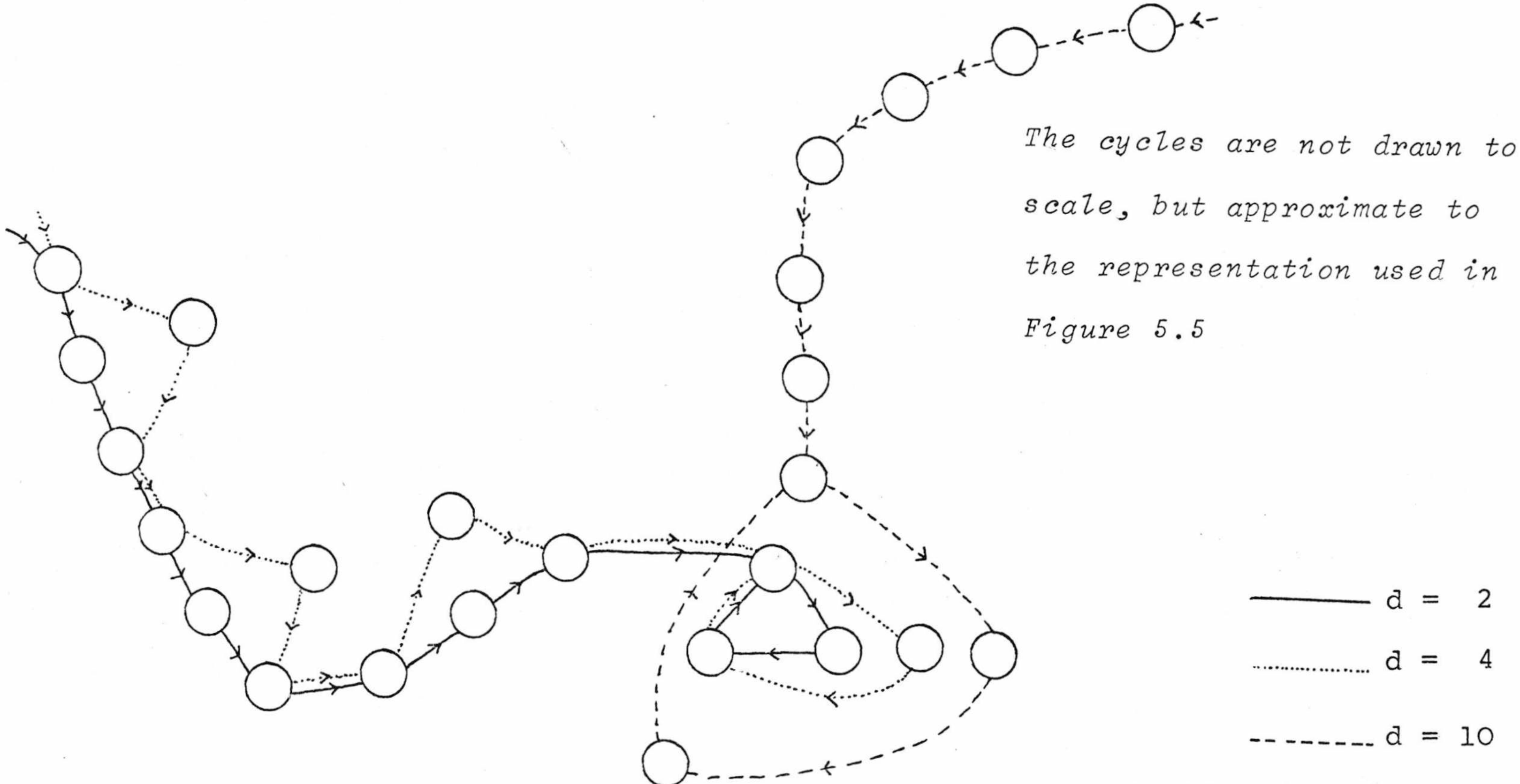
5.6 Discussion of results.

The results presented in the preceding sections demonstrate how the principle of contextual processing can be found in a network trained on a sequence of stimulus patterns. In section 5.4 it was shown that the system can "recognise" distorted versions of a learnt sequence by producing a response which is a sympathetic generation of



d = Hamming distance distortion with respect
to prototype.

Figure 5.6: Progressive distortion of middle frame.



d = distortion (Hamming distance) of frame B_n.

Figure 5.7: General state space dynamics.

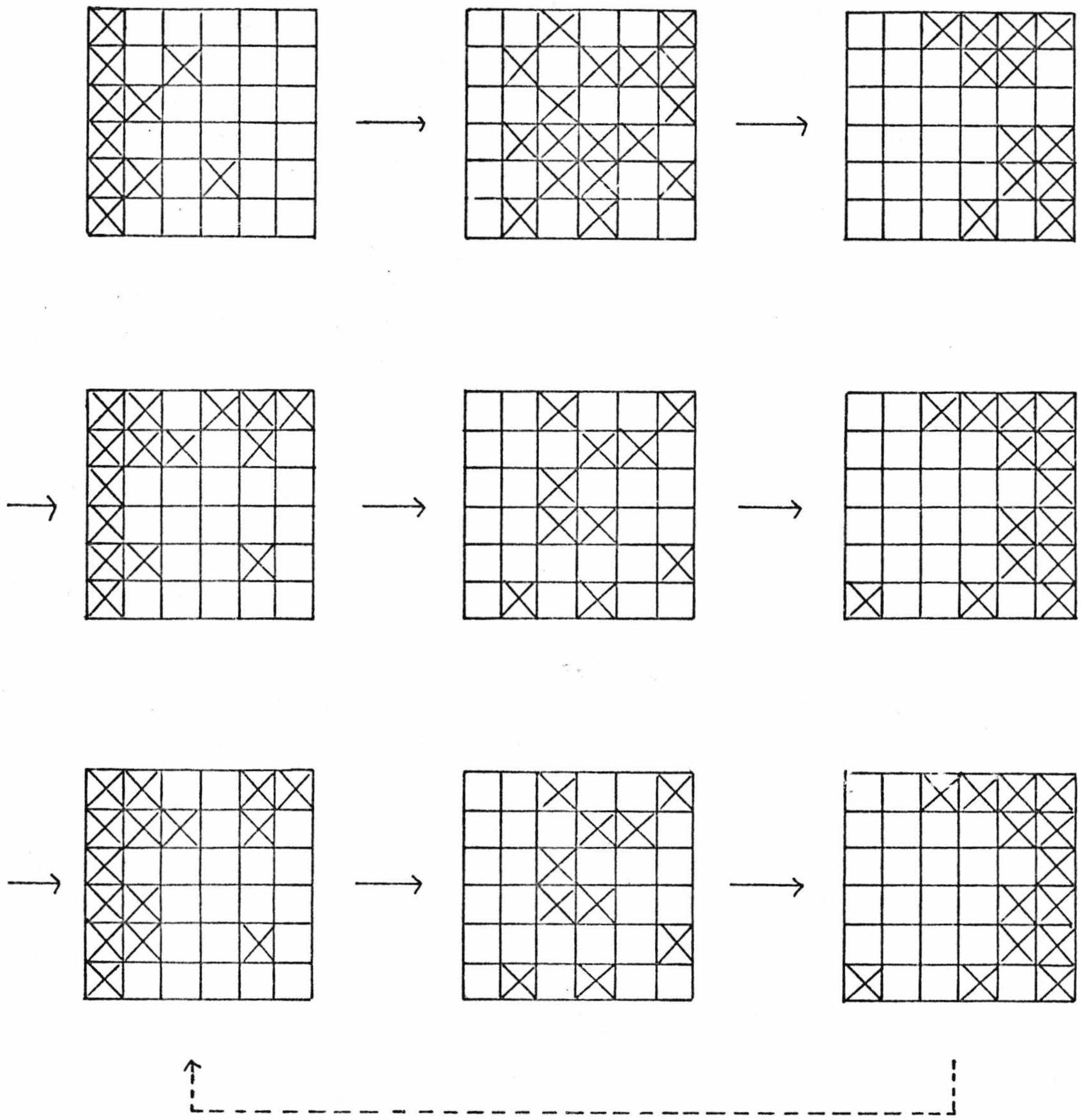


Figure 5.8: *Output sequence when input degraded by Hamming distance=12*

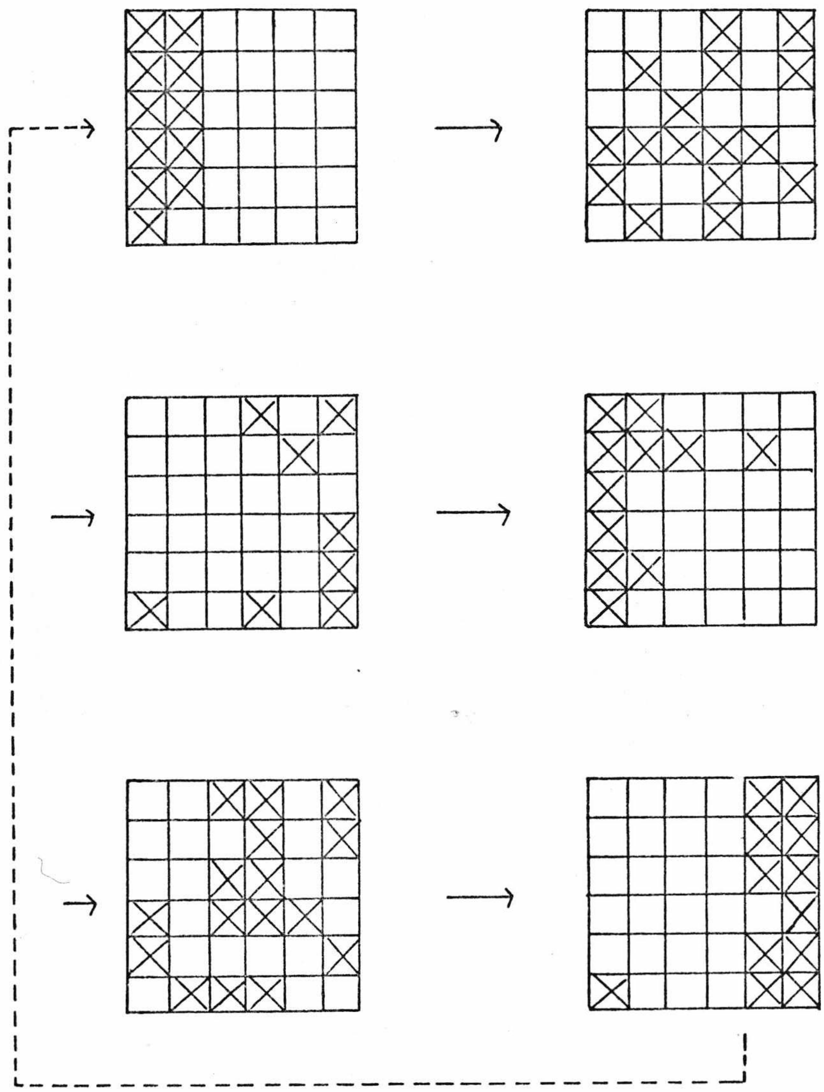


Figure 5.9: Sequence after breakdown in synchronism.

a similar output sequence. Just as with single frames, the net internally generates an "archetype" representation of the pattern class found in its training environment.

The use of context was illustrated further in section 5.5 by observing the response of the system when, after training on the generator sequence, the middle frame was progressively distorted by an increasing amount. Here it was found that a sympathetic cycling occurred even with large amounts of distortion introduced. The sequence was still "recognised" under conditions where, if the distorted frame were being considered as a single entity, it would probably have been classified in an indeterminate area of state space. The net, however, was able to "see" the frame in the context of the sequence and produce a response accordingly.

Evidence that the net responds to a set of inputs as a <u>sequence</u> rather than individual patterns can be further illustrated by presenting a previously learnt sequence to the trained net with the frames <u>in reverse order</u>. For a typical input, the response of the net is an output state sequence of 24 patterns, the last 3 states in this case (but not of necessity) forming a recurring cycle. These three states (see Figure 5.10) show less correspondence to the frames previously taught to the system. This corresponds to a situation where, for example, a person is

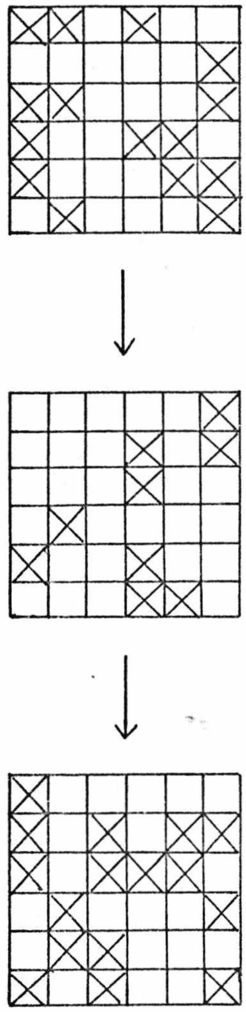


Figure 5.10: Output cycle when sequence presented in reverse order.

taught that the sequence $T \to H \to E$ is a meaningful word, whereas the reverse sequence $E \to H \to T$ is not. (It is interesting to note that, since the training sequence is a "closed" sequence (i.e. cyclic), the system would also "recognise" the input sequence $H \to E \to T$. This question of differences between "open" and "closed" sequences is one which needs to be investigated in the future).

5.7 Explanation of observed results.

The key to an explanation of the behaviour of the network in response to pattern sequences lies in the chaining effect of the OR-ed feedback connections as a new stimulus is applied with each successive clock pulse (see section 5.2). By looking at the pattern actually seen by the s.l.a.m. elements (i.e. the result of the OR operation between the stimulus and the feedback pattern) it becomes clear that the behaviour can be explained in familiar Hamming distance terms, and that no new mechanism is involved.

Consider, for example, the case where the net is trained on the sequence of stimuli

```
XX...          ..XX..          ....XX
XX...          ..XX..          ....XX
XX...          ..XX..          ....XX
XX...    →     ..XX..    →     ....XX
XX...          ..XX..          ....XX
XX...          ..XX..          ....XX

S(1)           S(2)            S(3)
```

.. .. (1)

The corresponding sequence of inputs actually seen by the s.l.a.m.s is then, by virtue of the OR-ed chaining

```
XX..XX         XXXX..          ..XXXX
XX..XX         XXXX..          ..XXXX
XX..XX    →    XXXX..    →     ..XXXX
XX..XX         XXXX..          ..XXXX
XX..XX         XXXX..          ..XXXX
XX..XX         XXXX..          ..XXXX

S(1)*          S(2)*           S(3)*
```

.. .. (2)

For each frame, the output S(n) is forced in response to S(n)* at the input to the s.l.a.m.s.

For a typical sequence of stimuli where the middle frame is degraded by a Hamming distance of 2, the input sequence seen by the s.l.a.m. elements is (as measured)

```
XX..XX          XXXX..          ..X.XX
XX..XX          XXXX..          ..XXXX
XX..XX    →     XXXX..    →     ..XXXX
XX..XX          XXXX..          ..XXXX
XX..XX          XXXX..          ..XXXX
XX..XX          XXXX..          ..XXXX
```

$S_2(1)*$ $S_2(2)*$ $S_2(3)*$

.. .. (3)

and if degraded by a Hamming distance of 8, the input sequence to the s.l.a.m.s becomes

```
XXX..X          XX.X..          ...XXX
XX..XX          XXXX.X          .X.XXX
XX..X.    →     X.XXX.    →     ..XXXX
XX..XX          XXXX..          ..XXXX
XX..XX          XXXXXX          ....XX
XX..XX          XXXX..          ..XXXX
```

$S_8(1)*$ $S_8(2)*$ $S_8(3)*$

.. .. (4)

Finally, if the training sequence is presented, but in reverse order, the inputs seen by the s.l.a.m. elements are

```
XX...X          XXXX.X          X...XX
XX.X.X          ..XX.X          X.X.XX
XX.X..    ←     XXXX.X    ←     X.XXXX
XX....          X.XXX.          .X..XX
XX.X..          X.XXXX          .XX.XX
XX.XX.          .XXX.X          X.X.XX
```

$S_B(1)*$ $S_B(2)*$ $S_B(3)*$

.. .. (5)

The situation is summarised in Figure 5.11, where the numbers within the boxes denote the Hamming distance between the sequence seen at the s.l.a.m. inputs during training (sequence 2) and the corresponding input sequence $S(1)^* \rightarrow S(2)^* \rightarrow S(3)^*$ seen by the s.l.a.m.s for each of the stimulus sequences S_2, S_8, S_B. It is seen that for the two cases where the test sequence is slightly degraded with respect to the training sequence (cases 3 and 4, corresponding to S_2 and S_8), the chaining effect produces small Hamming distances between test inputs and training inputs as actually seen by the s.l.a.m. elements.

When the training sequence is presented in reverse order (case 5, corresponding to sequence S_B) however, the input patterns seen by the s.l.a.m.s are much more dissimilar (large Hamming distances) to those seen in training. As a result the reverse sequence produces a meaningless set of output patterns as previously described.

The results outlined in this chapter provide evidence that the principles of contextual analysis have application in studies of digital learning nets. The use of such techniques is clearly significant, and greatly enhances the practical value of these networks.

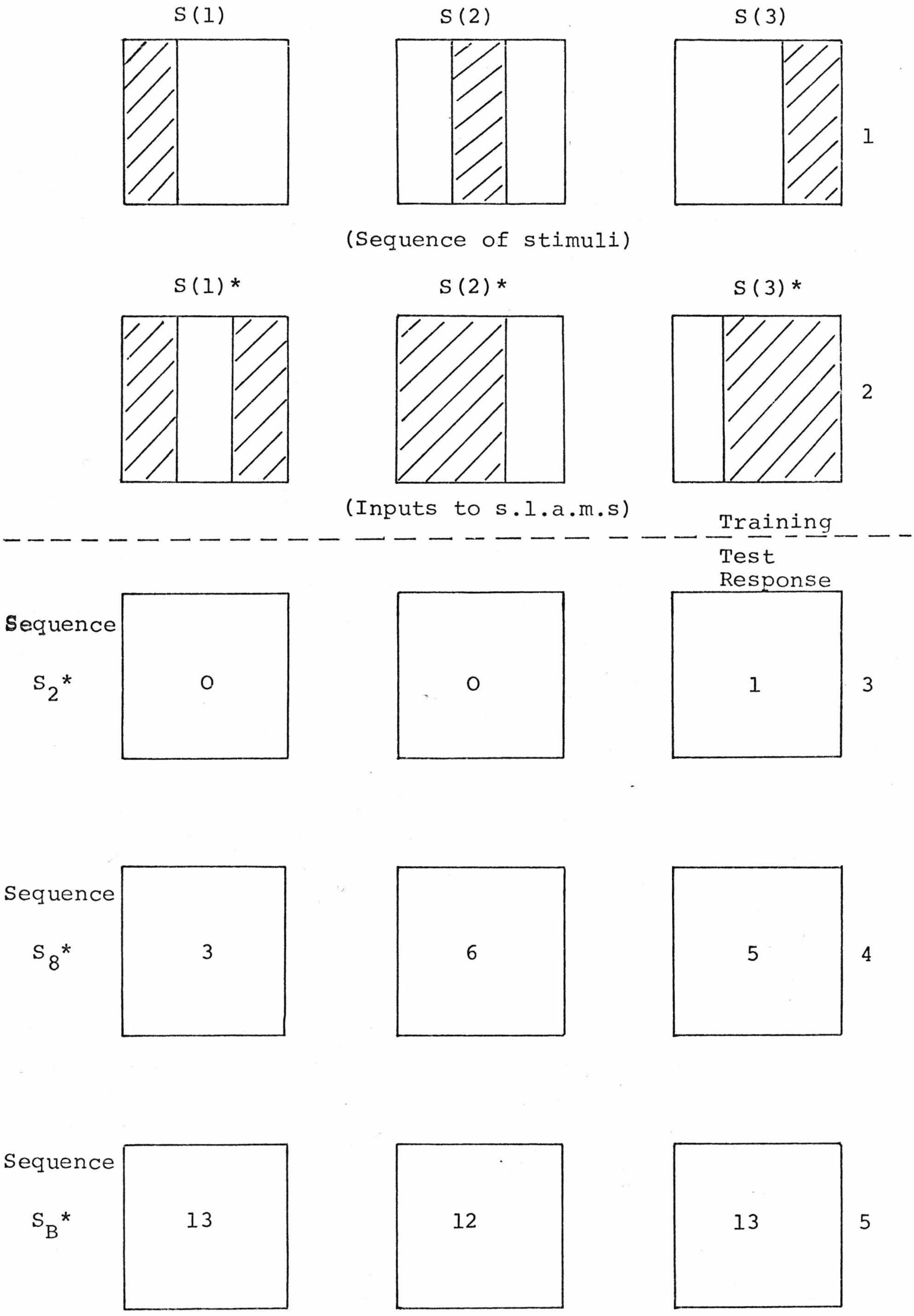


Figure 5.11: Hamming distance relations.

CHAPTER 6

ANALYSIS AND DISCUSSION

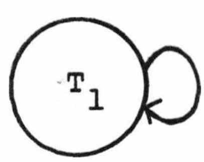
6.1 Background to the analysis.

Throughout the preceding chapters where experimental work has been discussed, an attempt has been made to explain and account for observed effects. In the main, this has been achieved by reference to specific examples rather than by a more general analysis. The following sections are intended to provide some analysis of the salient empirical observations which suggest a basis for a theory of dynamic learning nets.

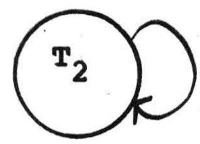
6.2 Interference of successive training patterns.

Let us consider a network of N elements which sample an input retina of N points, and let there be n inputs per element.

Let the net be trained on a single pattern T_1 with the teach terminals of the elements sampling the input retina. This situation will give rise to the recurring state cycle



If a second training pattern, T_2, is added, then a new cycle



will be formed, but with a resulting possibility that the original cycle due to T_1 is destroyed.

This disturbance of the T_1 cycle will occur when, in training, an element is required to produce a <u>different</u> output (for T_1 and T_2) for an input n-tuple (at a s.l.a.m.) which is the <u>same</u> for the two training patterns.

Let the Hamming distance between T_1 and T_2 be H_T. We wish to evaluate the probability that an n-tuple sampled on the input is the same for different forced outputs corresponding to pattern T_1 and T_2.

The total possible number of ways of changing H_T points on a retina of N points is

$$\binom{N}{H_T} .$$

The number of ways of changing H_T points from N such that an n-tuple sampled is the <u>same</u> for T_1 and T_2 is given by

$$\binom{N - n}{H_T} .$$

Therefore, the probability that an input n-tuple be the same for T_1 and T_2 is given by

$$Q = \frac{\binom{N - n}{H_T}}{\binom{N}{H_T}}$$

$$= \frac{(N-H_T)(N-H_T-1) \text{ --- } (N-H_T-n+1)}{N(N-1) \text{ --- } (N-n+1)}$$

$$\qquad \qquad .. \quad .. \quad .. \quad (6.1)$$

The probability that the output of this particular element should be <u>different</u> for T_1 and T_2 (as a result of training) is

$$\frac{H_T}{N} \; .$$

Therefore, the probability that the s.l.a.m. will be required to produce a different output for the same input n-tuple, is equal to the probability of a "<u>contradication</u>" in training, and is given by

$$P_c = \frac{H_T \, Q}{N} \qquad .. \quad .. \quad .. \quad (6.2)$$

Since there are N s.l.a.m.s, the most likely number of elements for which contradication occurs is given by

$$H_C = N \times P_C$$

$$= H_T Q \qquad (6.3)$$

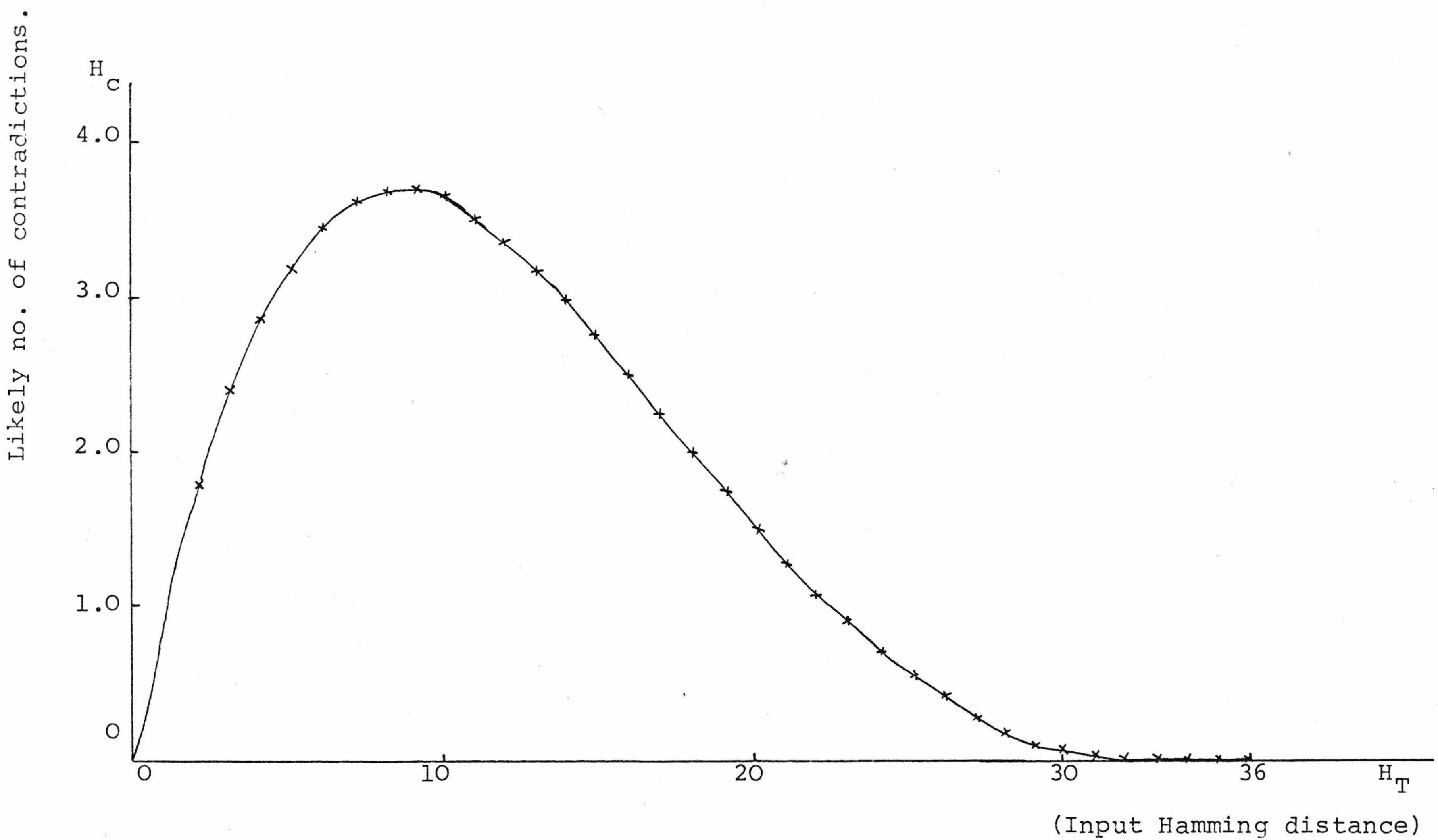
For each contradiction, the n-tuple feature of the first training pattern T_1 will be overwritten in favour of the corresponding feature of the later training pattern T_2.

Figure 6.1 shows a graph of the H_C against H_T for all values of H_T (for N=36 and n=3), illustrating the variation in the magnitude of the contradiction effect. The graph shows that the likely number of contradictions only falls below unity (i.e. no likely disturbance of an earlier trained cycle) for values of H_T such that

$$H_T \leqslant 1$$

$$\text{or} \quad H_T \geqslant 23$$

It is clear that for a group of very similar training patterns contradiction is likely to occur. When this happens the features of a later pattern dominate, thus disturbing the stable cycle formed by an earlier training pattern. The cycle formed by the last-seen training pattern then remains undisturbed and can draw in cycles arising from other training patterns. This was discussed qualitatively in section 2.5, where training patterns had values of H_T such that $1 \leqslant H_T \leqslant 4$.



<u>Figure 6.1</u>: Variation in "contradiction" effect.

Similarly, in section 2.6, where the net was trained on two pattern classes, typical patterns from each of the two groups were found to be at a Hamming distance of 20 apart on average. From the graph, this represents a probability that some contradictions will occur - this time between the two groups of patterns as well as between patterns within each group - and hence individual cycles of early training patterns were again found to be slightly disturbed.

6.3 Training with less than 100% sampling by teach wires.

If the number of active (i.e. clocked) teach signals is reduced from N (as in section 6.2) to W when the second training pattern T_2 is presented, then as in Equation 6.2

$$P_c = \frac{H_T \, Q}{N}$$

But now the probability that the teach signal to a particular element be clocked is

$$\frac{W}{N} .$$

The probability of disturbance of the earlier T_1 cycle is therefore given by

$$P_c^W = \frac{H_T \, Q \, W}{N^2} \qquad \ldots \ldots \ldots \qquad (6.3)$$

Similarly, the most likely number of contradictions is given by

$$H_C^W = \frac{H_T\ Q\ W}{N} \qquad \text{.. .. ..} \qquad (6.4)$$

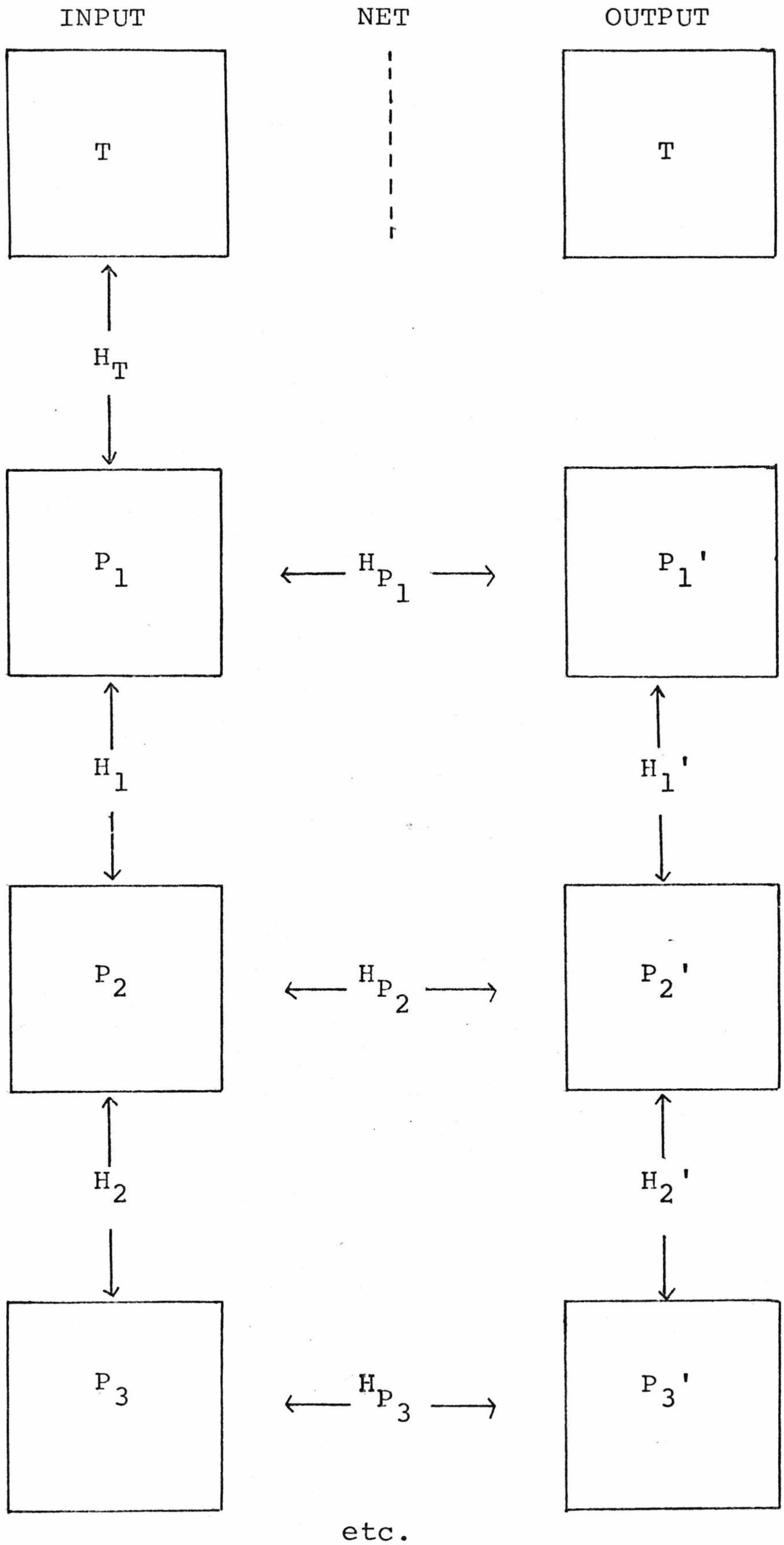
For $H_T = 4$, and using the decreasing teach activity described in section 3.4, the values of H_C^W shown in Table 6.1 are obtained. It is seen that the probability of contradiction occurring decreases markedly when teach activity is reduced.

Number of active teach wires (W)	H_C^W
36	2.79
18	1.39
9	0.70
5	0.39
2	0.15

Table 6.1

6.4 Unconstrained net with feedback.

This section considers the cycling activity of a net where the output at any instant θ directly becomes the next input at time θ+1. The notation used for multiple passes through the net in this and the following sections is defined in Figure 6.2.

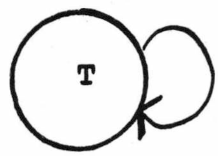


Training pattern is T Test stimulus is P_1

H = Hamming distance

Figure 6.2: Notation.

Suppose the net is trained on a pattern T. Then, as before, this gives rise to a cycle



Now consider the response of the net to an input pattern P_1, where the Hamming distance between P_1 and T is H_T. Let the net consist of N elements, each with n inputs, sampling an N-point retina as before.

Then, as derived in Equation 6.1, the probability that an input n-tuple be the <u>same</u> for T and P_1 is given by

$$Q = \frac{(N-H_T)(N-H_T-1) \text{ --- } (N-H_T-n+1)}{N(N-1) \text{ --- } (N-n+1)}$$

Also, the probability of corresponding points on P_1 and T having different values is

$$\frac{H_T}{N} .$$

In other words, the probability of two corresponding points being the same is

$$1 - \frac{H_T}{N}$$

Therefore, the probability that an element should sample the same n-tuple for P_1 and T <u>and</u> that the output of this element have the same value as the corresponding point on the P_1 <u>input</u> matrix, is

$$Q \left[1 - \frac{H_T}{N}\right] .$$

This is also the probability that as a result of common addressing, an output point on P_1' shall have the same value as the corresponding point on P_1 (see Figure 6.2 for notation).

And hence the probability of corresponding points on P_1 and P_1' being different is

$$1 - Q \left[1 - \frac{H_T}{N}\right] .$$

In addition, if the s.l.a.m. stores are initially randomly half-filled with logical ones, there is a 50% chance that locations addressed by <u>different</u> n-tuples will also produce the same value as corresponding points on P_1.

Therefore, the overall probability that an output point be different from the corresponding point on the input is

$$\frac{1}{2}\left[1 - Q\left(1 - \frac{H_T}{N}\right)\right].$$

Then, correspondingly, the probable Hamming distance (H_{P_1}) between P_1 and P_1' is given by

$$H_{P_1} = \frac{N}{2}\left[1 - Q\left(1 - \frac{H_T}{N}\right)\right]. \qquad .. \quad .. \quad .. \quad (6.5)$$

On feeding back, this output becomes the next input to the net directly (i.e. $P_2 = P_1'$).

Hence, the probability that an element shall have the same input sample (and therefore the same output) for P_1 and P_2 is given by

$$Q' = \frac{\binom{N-n}{H_{P_1}}}{\binom{N}{H_{P_1}}} \qquad .. \quad .. \quad .. \quad (6.6)$$

(following once again the argument used in deriving Equation 6.1).

As before, there is also a 50% chance that elements sampling <u>different</u> n-tuples will produce the same value at a point on P_2' as occurred on P_2.

Therefore, the probability that a point on $P_2{}'$ be different from the corresponding point on P_2 is

$$\frac{1}{2}\left[1 - Q'\right] .$$

And hence, from this

$$H_{P_2} = \frac{N}{2}\left[1 - Q'\right] \qquad \ldots \ldots \ldots (6.7)$$

Equation 6.6 and Equation 6.7 therefore provide an iterative procedure whereby successive transition Hamming distances can be predicted by evaluating H_{P_n} (Equation 6.7) and substituting back into Equation 6.6 to find the next value of $Q^{(n)}$.

For some typical values used in the experimental work we take N=36, n=3, and H_T=4. The predicted sequence of transition Hamming distances is then given by

$$H_{P_1} \approx 7$$

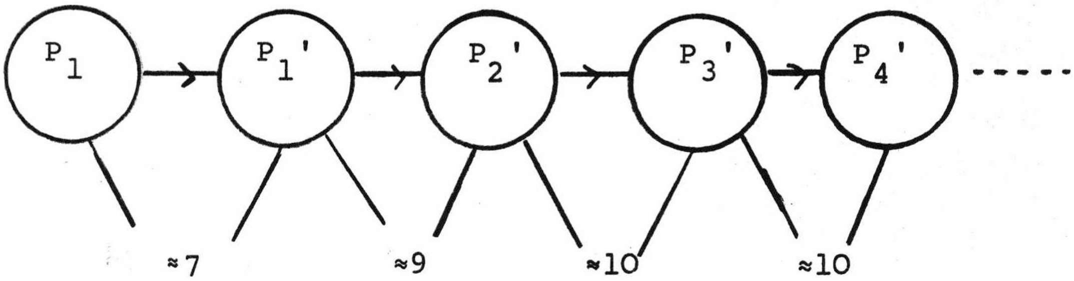
$$H_{P_2} \approx 9$$

$$H_{P_3} \approx 10$$

$$H_{P_4} \approx 10$$

etc.

i.e. the state sequence is



These predicted Hamming distances are large and show a divergence effect. This would be expected to give rise to fairly long cycles of states, which is the behaviour actually observed and reported in Chapter 2.

6.5 Net constrained by means of OR-ed feedback connections.

Now consider the net trained on the same pattern as that in section 6.4, but this time observe the response when the pattern in the feedback path is OR-ed with the applied stimulus pattern to produce the next input to the net.

For the first output, the analysis follows that given in section 6.4, so that the probable Hamming distance between P_1 and P_1' is once again given by

$$H_{P_1} = \frac{N}{2}\left[1 - Q\left(1 - \frac{H_T}{N}\right)\right] \qquad \text{.. .. .. (6.8)}$$

where Q is given by Equation 6.1 as before.

On clocking the feedback loop, the output is OR-ed with the original stimulus, so that

$$P_2 = P_1 + P_1' .$$

(This was explained formally in section 3.1).

Now suppose that there are 'a' logical "ones" on the pattern P_1 (the stimulus).

Then the probability that a point on P_2 be a logical zero is

$$\frac{N - a}{N} . \qquad \text{.. .. .. (6.9)}$$

Because of the nature of the OR function (see Table 6.2), the only points on P_2 which can be <u>different</u> from corresponding points on P_1 are those which are logical zero on P_1 <u>and</u> logical one on P_1'.

I_1	I_2	$F = I_1 + I_2$
0	0	0
0	1	1
1	0	1
1	1	1

+ denotes logical OR

Table 6.2

Therefore, the probability that a point on P_1 be different from the corresponding one on P_1' (due to H_{P_1}), and that it is zero on P_1 is

$$\frac{(N - a)}{2N} \left[1 - Q\left(1 - \frac{H_T}{N}\right)\right] \quad . \qquad \ldots \ldots (6.10)$$

(This is derived from Equation 6.8 and Equation 6.9).

Then the probable Hamming distance between P_1 and P_2 ($= P_1 + P_1'$) is H_1 and is given by

$$H_1 = \frac{(N - a)}{2} \left[1 - Q\left(1 - \frac{H_T}{N}\right)\right] . \qquad \ldots \ldots \ldots (6.11)$$

The probability of an element sampling the <u>same</u> n-tuple for P_1 and P_2 is Q', which is given by

$$Q' = \frac{\binom{N - n}{H_1}}{\binom{N}{H_1}}$$

Hence, the probability that element outputs be different for corresponding points on P_1' and P_2' (following the argument given in section 6.4) is

$$\frac{1}{2}\left[1 - Q'\right] \qquad .. \quad .. \quad .. \quad (6.12)$$

Then the likely Hamming distance (H_1') between P_1' and P_2' (the first two successive outputs from the net) is given by

$$H_1' = \frac{N}{2}\left[1 - Q'\right] \qquad .. \quad .. \quad .. \quad (6.13)$$

The probability of corresponding points on P_1' and P_2' being different is given by Equation 6.12.

Therefore, the probability that a point on P_1' be different from the corresponding point on P_2' <u>and</u> that it should fall outside the 'a' ones on the stimulus pattern P_1 is

$$\frac{(N - a)}{N} \times \frac{1}{2}\left[1 - Q'\right].$$

Hence, the probable Hamming distance (H_2) between the next successive <u>inputs</u> to the net (P_2 and P_3) is given by

$$H_2 = \frac{(N - a)}{2} \left[1 - Q' \right] \quad \ldots \ldots \quad (6.14)$$

Equations 6.13 and 6.14 provide an iterative procedure to predict the transition Hamming distance values for successive cycles of the net with OR-ed feedback connections.

Taking the same values of N (=36), n (=3), and H_T (=4) as in section 6.4, the following predicted transition Hamming distance values are obtained.

$H_1 \approx 7$

$H_2 \approx 4$

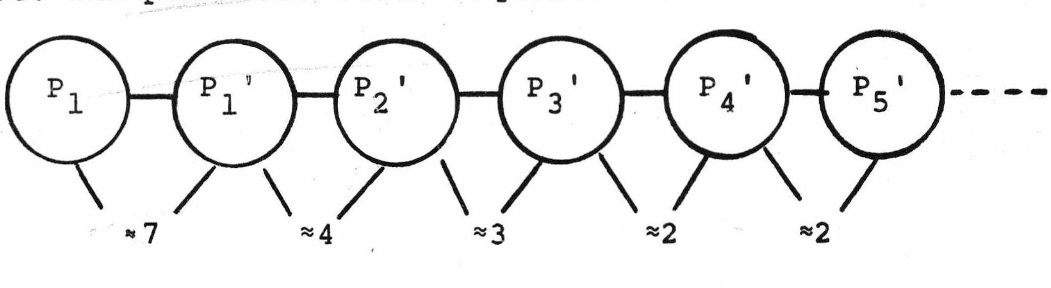
$H_3 \approx 3$

$H_4 \approx 2$

$H_5 \approx 2$

etc.

i.e. the predicted state sequence is



These calculations predict a convergence of Hamming distance values, which is likely to result in shorter cycles than in the case of unrestricted feedback (see section 6.4). This is precisely the behaviour which is observed in the net with OR-ed feedback connections which was described in Chapter 3.

6.6 Concluding summary.

Since detailed conclusions and discussions have been considered in the body of the thesis, only a brief summary is included at this point.

It was stated in Chapter 1 that one stimulus to the present work was the concept of functional brain modelling using the established techniques of automata theory. In the light of this it is felt that the main contribution of this thesis has been to investigate the nature of the state cycles occurring in the network, representing what might be called its "thought processes"*, and in particular it has been shown how an unsupervised net learns to form an internal model of the environment in which it "lives".

Recognition and classification of input stimuli can then be defined in terms of the state cycling of the

* *Cycles and "thought processes" have been related in this way by Caianiello (see Reference 32).*

dynamic system. This definition has also been extended to include the case where the stimulus comprises a <u>sequence</u> of individual patterns. The nature of memory in the net has been discussed and the distinction between short-term and long-term memory processes accounted for.

More generally, in the broader field of artificial intelligence, the work has produced some empirical results in a field hitherto largely unexplored, namely the investigation of networks of logic elements where functions are not random, but are <u>learned</u> as a result of exposure to a non-random environment.

The research discussed here opens several avenues for further work. In particular, two points require investigation in the immediate future.

<u>Point (i)</u> is the question of discovering the limit to the permissible number of training classes.

The existence of distinct state space regions has been established for two training classes, but the general problem of multi-class learning has not been tackled.

<u>Point (ii)</u> is the problem of applying the techniques developed in the present work to larger networks.

This is obviously related to Point (i) but would have a wider relevance in so far as it would permit processing of "real" patterns (i.e. actual handwritten characters, suitably quantised and coded). The use of such data would allow an assessment of the practical value of the system to be made.

In practical terms it may prove necessary to look beyond the present system to a multi-layer network, where features extracted by one net are used as inputs to a second layer of elements which can be trained separately. Each output is therefore influenced by a large number of inputs but the number of functions which can be performed is restricted.

The theoretical analysis presented in this chapter has been confined to a general treatment of the problem of interference between training patterns when 'teach' information is extracted from sensory data, and to predicting the dynamic response of a net in cases of restricted and unrestricted cycling. In the future it is hoped that the theoretical basis which has been established here will be extended to the problems outlined in Points (i) and (ii) above.

APPENDIX I

FUNCTIONAL DESCRIPTION OF THE S.L.A.M. ELEMENT

Figure A1.1 shows a block diagram of a s.l.a.m.-8 (i.e. 3-input) element.

The input to the device (x_1, x_2, x_3) is decoded to excite one out of the eight possible address lines. In a "teach" phase a teach clock input (c) is activated to enable the **desired** value of the output to be written into the appropriate memory location (the desired output is transmitted via the "teach" input, T). The output (f) of the device is the logical state (ϕ) of the location currently being addressed by the input pattern $x_1\ x_2\ x_3$.

Functionally, the 3-input element can be described by the equation

$$f = \phi_0\ \bar{x}_1\ \bar{x}_2\ \bar{x}_3 + \phi_1\ \bar{x}_1\ \bar{x}_2\ x_3 + \phi_2\ \bar{x}_1\ x_2\ \bar{x}_3$$

$$\text{-----}\quad \phi_7\ x_1\ x_2\ x_3$$

These are sequential equations where

$$\phi_0' \quad \bar{x}_1\bar{x}_2\bar{x}_3\ T\ c + \phi_0\ \bar{c}$$

$$\cancel{\phi_0' = \bar{x}_1\ \bar{x}_2\ T\ c + \phi_0\ \bar{c}}$$

$$\phi_1' = \bar{x}_1\ \bar{x}_2\ x_3\ T\ c + \phi_1\ \bar{c}$$

etc.

(ϕ_n is the present state of a memory location and ϕ_n' its next state).

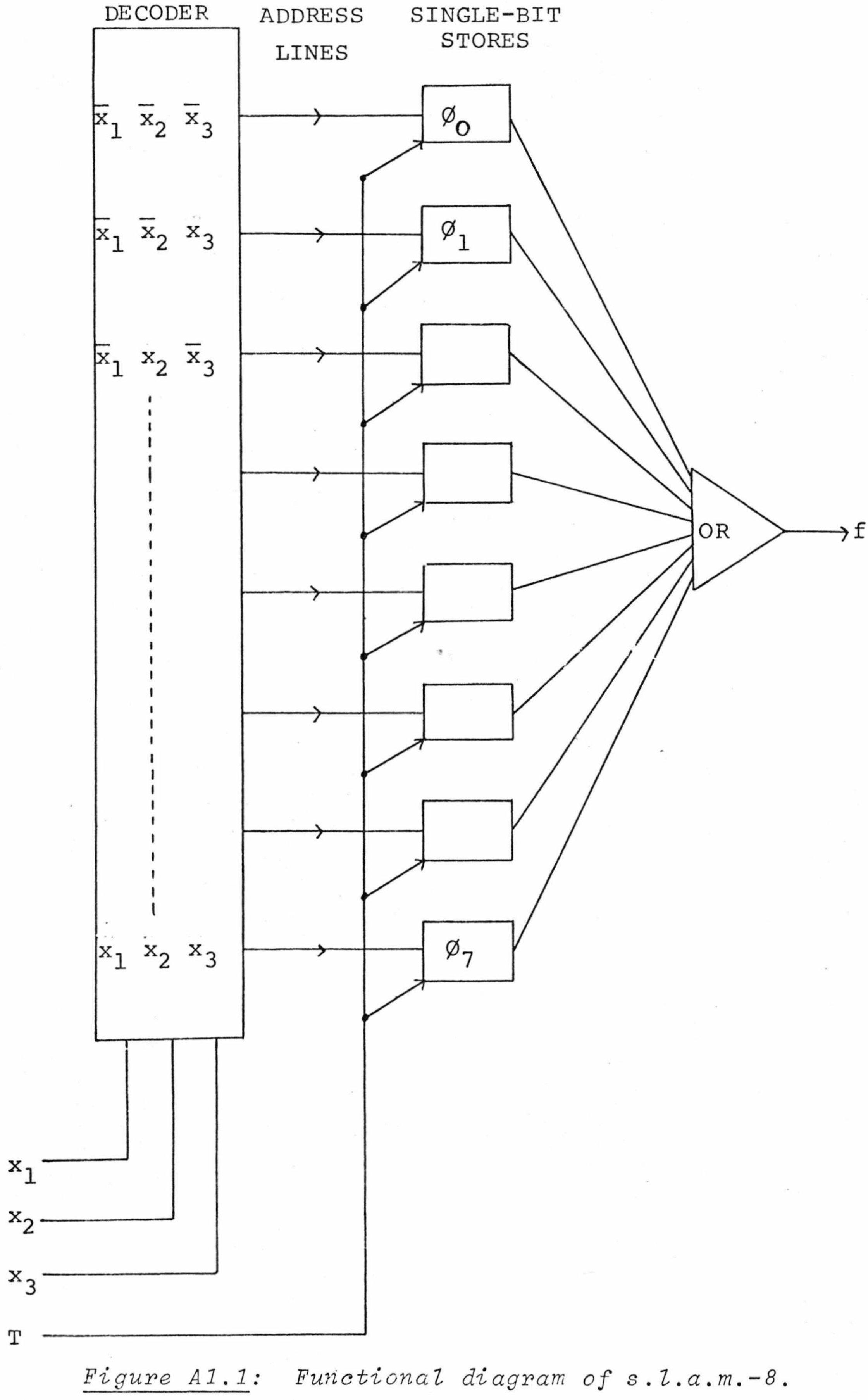


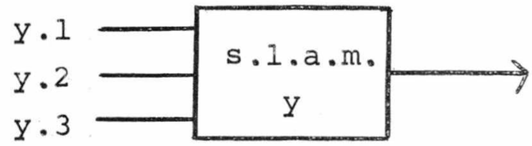
Figure A1.1: *Functional diagram of s.l.a.m.-8.*

APPENDIX II

RANDOM INTERCONNECTIONS

In the simulated networks, the inputs to the s.l.a.m. elements are connected randomly to the input matrix.

Let the s.l.a.m. inputs be labelled as follows



and let the input matrix points be identified as follows

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17
18	19	20	21	22	23
24	25	26	27	28	29
30	31	32	33	34	35

Then the actual connections used in the simulation are given in Table A2.1.

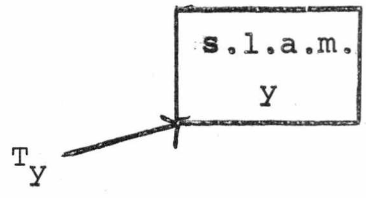
s.l.a.m. input	Matrix point	s.l.a.m. input	Matrix point	s.l.a.m. input	Matrix point
1.1	20	13.1	6	25.1	6
1.2	28	13.2	14	25.2	27
1.3	0	13.3	24	25.3	12
2.1	29	14.1	5	26.1	35
2.2	6	14.2	2	26.2	20
2.3	35	14.3	3	26.3	7
3.1	8	15.1	22	27.1	28
3.2	17	15.2	31	27.2	4
3.3	24	15.3	29	27.3	9
4.1	9	16.1	10	28.1	30
4.2	21	16.2	19	28.2	13
4.3	15	16.3	33	28.3	5
5.1	2	17.1	0	29.1	29
5.2	4	17.2	15	29.2	2
5.3	19	17.3	4	29.3	18
6.1	3	18.1	7	30.1	14
6.2	22	18.2	23	30.2	8
6.3	1	18.3	8	30.3	23
7.1	30	19.1	27	31.1	21
7.2	12	19.2	21	31.2	11
7.3	32	19.3	34	31.3	26
8.1	10	20.1	11	32.1	22
8.2	5	20.2	30	32.2	1
8.3	13	20.3	13	32.3	16
9.1	16	21.1	17	33.1	19
9.2	25	21.2	12	33.2	33
9.3	18	21.3	26	33.3	10
10.1	11	22.1	25	34.1	31
10.2	31	22.2	32	34.2	0
10.3	26	22.3	35	34.3	15
11.1	33	23.1	18	35.1	25
11.2	23	23.2	28	35.2	32
11.3	34	23.3	16	35.3	17
12.1	14	24.1	9	36.1	24
12.2	7	24.2	1	36.2	34
12.3	27	24.3	20	36.3	3

Table A2.1: Element connections.

APPENDIX III

DECREASING TEACH ACTIVITY

Let the "Teach" inputs to the s.l.a.m.s be labelled as follows



Then, in the experiments where teach activity was progressively decreased, the teach signals were applied to the s.l.a.m.s according to Table A3.1

Total number active	s.l.a.m.s to which teach signals applied
36	All
18	2, 6, 7, 10, 12, 14, 15, 18, 19,20,21, 22, 26, 29, 30, 31, 32,34
9	2, 6, 7, 10, 15, 18, 21,26, 30
5	15, 18, 21, 26,30
2	15, 30

Table A3.1

APPENDIX IV

A BRIEF NOTE ON THE SIMULATIONS

The experiments on s.l.a.m. networks described in the text were carried out entirely by software simulation, rather than by hardware construction. The simulations were written in the DAP-16 assembly language and the programs run on a Honeywell DDP-516 computer (16-bit word, initially 8K store but later increased to 16K).

The experimental data (T-patterns and H-patterns etc.) was stored on magnetic tape and referenced by means of a simple identifying code which indicated

(a) The generator from which a particular pattern was derived.

(b) Its Hamming distance with respect to that generator.

(c) Its own individual reference number.

The output of the net was recorded in the form of either a visual print-out on a teletype, or punched paper tape for further processing, or both.

REFERENCES

1. Bowden, B.V. : "Faster than Thought"
New York; Pitman 1953

2. Turing, A.M. : "Computing machinery and intelligence"
Mind, Vol.59, 433-460, 1950

3. Armer, P. : "Attitudes toward intelligent machines"
Symposium on Bionics, Wadd Technical Report 60 600, 13-19, 1960

4. Newell, A., Shaw, J.C., & Simon, H. : "Chess playing programs and the problem of complexity"
IBM J. of Res. & Dev., Vol.2, 320-335, 1958

5. Samuel, A.L. : "Some studies in machine learning using the game of checkers"
IBM J. of Res. & Dev., Vol.3, 211-229, 1959

6. Tonge, F.M. : "Assembly Line Balancing Using Probabilistic Combinations of Heuristics"
Management Sci., Vol.11, 727-35, 1965

7. Buchanan, B.,
Sutherland, G.,
& Feigenbaum, E.A. : "Heuristic Dendral : a program for generating explanatory hypotheses in organic chemistry"
In "Machine Intelligence 4"
(Eds. Meltzer & Michie)
Edinburgh Univ. Press 1969

8. Slagle, J.R. : "Artificial Intelligence : The heuristic programming approach"
McGraw-Hill, 1971

9. Fogel, L.J.,
Owens, A.J.,
& Walsh, M.J. : "Artificial intelligence through simulated evolution"
Wiley, 1966

10. Minsky, M. : "Steps toward artificial intelligence"
Proc. I.R.E., Vol.49, 8-30, 1961

11. Solomonoff, R. : "Some recent work in artificial intelligence"
Proc. IEEE, Vol.54, 1966

12. Feigenbaum, E.A. : "Artificial intelligence : Themes in the second decade" In A.J.H. Morrell (Ed.) "Information Processing 68" Vol.2, 1008-1022, North-Holland Publishing Co., Amsterdam, 1969

13. Pask, G. : "A discussion of artificial intelligence and self-organisation" Advances in Computers, Vol.5, 110-226, 1964

14. Nilsson, N.J. : "Learning machines" McGraw-Hill, 1965

15. Bongard, M.M. : "Pattern recognition" Spartan Books, 1970

16. Watanabe, S. (Ed.) : "Methodologies of pattern recognition" Academic Press, 1969

17. Travis, L.E. : "The value of introspection to the designer of mechanical problem solvers" Behav. Sci., Vol.8, 227-33, 1963

18. Wason, P.C. & Johnson-Laird, P.N. (Eds.) : "Thinking and Reasoning" Penguin Books, 1968

19. Kolers, P.A. : "Recognizing patterns : Studies in living and automatic systems" M.I.T. Press, 1968

20. von Neumann, J. : "The computer and the brain" Yale Univ. Press, New Haven, 1958

21. George, F.H. : "The brain as a computer" Pergamon Press, 1961

22. Wooldridge, D.E. : "The machinery of the brain" McGraw-Hill, 1963

23. Hebb, D.O. : "The organisation of behaviour : A neuropsychological theory" Wiley, 1949

24. Farley, B.G., & Clark, W.A. : "Simulation of self-organising systems by digital computer" Trans. IRE, Vol.PGIT-4, 76-84, 1954

25. Rochester, N., Holland, J.H., Harbt, L.H., & Duda, W.L. : "Tests on a cell assembly theory of the action of the brain, using a large digital computer" Trans. IRE, Vol.PGIT-2, 80, 1956

26. McCulloch, W.S., & Pitts, W. : "A logical calculus of the ideas immanent in nervous activity" Bull. Math. Biophysics, Vol.5, 115-133, 1943

27. Brazier, M.A.B. : "The electrical activity of the nervous system" Pitman, 1951

28. Rosenblatt, F. : "Principles of neurodynamics" Spartan Books, 1962

29. Minsky, M., & Papert, S. : "Perceptrons : an introduction to computational geometry" M.I.T. Press, 1969

30. Widrow, B., & Hoff, M.E. : "Adaptive Switching Circuits" Wescon Convention Record, IRE, 1961

31. Taylor, W.K. : "Pattern recognition by means of automatic analogue apparatus" Proc. I.E.E., Vol.106B, 198-209, 1959

32. Caianiello, E.R. : "Outline of a theory of thought-processes and thinking machines" J. Theoret. Biol., Vol.2, 204-35, 1961

33. Caianiello, E.R. : "Decision equations and reverberations" Kybernetik, Vol.3, 98-100, 1966

34. Aleksander, I. : "Adaptive systems of logic networks and binary memories" Proc. Spring Joint Comp. Conf., 707-12, 1967

35. Aleksander, I., & Albrow, R.C. : "Adaptive logic circuits" Computer Journal, Vol.11, 65-71, 1968

36. Aleksander, I., & Albrow, R.C. : "Pattern recognition with adaptive logic elements" Proc. I.E.E.-NPL Conf. on Pattern Recognition.IEE pub. no. 42, 1-10, 1968

37. Aleksander, I., & Fairhurst, M.C. : "Pattern learning in humans and electronic learning nets" Electron. Lett., Vol.6, 518, 1970*

38. Aleksander, I. : "Microcircuit learning nets : Hamming distance behaviour" Electron. Lett., Vol.6, 139, 1970

39. Epstein, W. : "Varieties of perceptual learning" McGraw-Hill, 1967

40. Baron, R.J. : "A model for the elementary visual networks of the human brain" Int. J. Man-Machine Studies, Vol.2, 267-90, 1970

41. Kabrisky, M. : "A proposed model for visual information processing in the human brain" University of Illinois Press, 1966

42. Noton, D. : "A theory of visual pattern perception" IEEE Trans. on System Science & Cybernetics, Vol.SSC-6, 349-357, 1970

43. Sparkes, J.J. : "Pattern recognition and a model of the brain"
Int. J. Man-machine Studies, Vol.1, 263-278, 1969

44. Hubel, D.H., & Wiesel, T.N. : "Receptive fields, binocular interaction and functional architecture in the cat's visual centre"
J. Physiol., Vol.160, 106-154, 1962

45. Lettvin, J.Y., Maturana, H.R., McCulloch, W.S., & Pitts, W.H. : "What the frog's eye tells the frog's brain"
Proc. IRE, Vol.47, 1940-1951, 1959

46. Wathen-Dunn, W. (Ed.) : "Models for the perception of speech and visual form"
M.I.T. Press, 1967

47. Aleksander, I. : "Automata theory and the computer scientist"
Computer Bulletin, Vol.14, 274-277, 1970

48. Booth, T.L. : "Sequential machines and automata theory"
Wiley, 1968

49. Arbib, M.A. : "Theories of abstract automata"
Prentice-Hall, 1969

50. Aleksander, I., & Fairhurst, M.C. : "An automaton with brain-like properties"
Kybernetes, Vol.1, 11-18, 1972*

51. Mamdani, E.H. : "Sequential adaptive logic circuits"
Ph.D. Thesis, University of London, 1971

52. Kiss, G.R. : "Networks as models of word storage"
In Collins & Michie (Eds.) "Machine Intelligence 1" Oliver & Boyd 1967

53. Kiss, G.R. : "Steps towards a model of word selection"
In Meltzer & Michie (Eds.) "Machine Intelligence 4" Edinburgh Univ. Press, 1969

54. Kauffman, S.A. : "Metabolic stability and epigenesis in randomly constructed genetic nets"
J. Theoret. Biol., Vol.22, 437-467, 1969

55. Hilgard, E.R. : "Theories of learning"
Appleton-Century-Crofts, Inc.
1948

56. Borger, R., & Seaborne, A.E.M. : "The psychology of learning"
Penguin Books, 1966

57. Bledsoe, W.W. & Browning, I. : "Pattern recognition and reading by machine"
Proc. Eastern Joint Comp. Conf., 225-232, 1959

58. Ullman, J.R. & Kidd, P.A. : "Recognition experiments with typewritten numerals from envelopes in the mail"
Pattern Recognition, Vol.1, 273-289, 1969

59. Ullman, J.R. : "Experiments with the n-tuple method of pattern recognition"
IEEE Trans. on Comp., Vol.18, 1135-38, 1969

60. Hull, C.L. : "A behaviour system"
Yale Univ. Press, 1952

61. Ashby, W.R. : "An introduction to cybernetics"
Chapman & Hall, 1956

62. Ashby, W.R. : "Design for a brain"
Chapman & Hall, 1952

63. Young, J.Z. : "A model of the brain"
Oxford Univ. Press, 1964

64. MacKay, D.M. : "Towards an information-flow model of human behaviour"
Brit. J. Phil., Vol.47, 30-43, 1956

65. Thomas, H.B.G. : "On the feedback regulation of choice behaviour"
Int. J. Man-machine Studies, Vol.2, 235-66, 1970

66. Annett, J. : "Feedback and human behaviour"
Penguin Books, 1969

67. Church, R.M., & Camp, D.S. : "Change in reaction time as a function of knowledge of results"
Am. J. Psych., Vol.78, 102-6, 1965

68. Nagy, G. : "State of the art in pattern recognition"
Proc. IEEE, Vol.56, 836-62, 1968.

69. Spragins, J. : "Learning without a teacher"
IEEE Trans. on Inf. Th.,
Vol.12, 223-229, 1966

70. Fralick, S.C. : "Learning to recognise patterns without a teacher"
IEEE Trans. on Inf. Th.,
Vol.13, 57-64, 1967

71. Lowenstein, O. : "The senses"
Penguin Books, 1966

72. Mellon, D. : "The physiology of sense organs"
Oliver & Boyd, 1968

73. Aleksander, I. : "Some psychological properties of digital learning nets"
Int. J. Man-machine Studies,
Vol.2, 189-212, 1970

74. Aleksander, I., & Mamdani, E.H. : "Microcircuit learning nets : improved recognition by means of pattern feedback"
Electron. Lett., Vol.4, 425, 1968

75. Neisser, U., & Weene, P. : "A note on human recognition of hand-printed characters"
Inf. & Control, Vol.3, 191-6, 1960

76. Aleksander, I., & Albrow, R.C. : "Microcircuit learning nets : some tests with handwritten numerals"
Electron. Lett., Vol.4, 408, 1968

77. Wilkins, B.R., & Ford, N.L. : "The analysis of training sets for adaptive pattern classifiers"
In "Machine perception of patterns and pictures"
Institute of Physics, 1972

78. Fairhurst, M.C. & Aleksander, I. : "Natural pattern clustering in digital learning nets"
Electron. Lett., Vol.7, 724-726, 1971*

79. Tryon, R.C. : "Cluster analysis"
Am. Arbor, Mich.: Edwards, 1939

80. Bonner, R.E. : "On some clustering techniques"
IBM J. of Res. & Dev., Vol.8, 22-32, 1964

81. Jardine, N. : "A new approach to pattern recognition"
Nature, Vol.234, 526-528, 1971

82. Baker, J. : "Pattern recognition in s.l.a.m. learning networks"
Internal Project Report, Electronics Laboratories, University of Kent, 1972

83. Fairhurst, M.C., & Aleksander, I. : "Dynamics of the perception of patterns in random learning nets"
In "Machine perception of patterns and pictures"
Institute of Physics, 1972 *

84. Slamecka, N.J.(Ed.): "Human learning and memory"
Oxford Univ. Press, 1967

85. Norman, D.A. : "Toward a theory of memory and attention"
Psychol. Rev., Vol.75, 522-36, 1968

86. Waugh, N.C., & Norman, D.A. : "Primary memory"
Psychol. Rev., Vol.72, 89-109, 1965

87. Deutsch, S. : "Models of the nervous system"
Wiley, 1967

88. Miller, G.A. : "Decision units in the perception of speech"
IRE Trans. on Inf. Theory, Vol.8, 81-83, 1962

89. Morton, J. : "Interaction of information in word recognition"
Psychol. Rev., Vol.76, 165-178, 1969

90. Abend, K. : "Compound decision procedures for pattern recognition"
Proc. NEC, Vol.22, 777-780, 1966.

91. Duda, R.O., & Hart, P.E. : "Experiments in the recognition of hand-printed text: Part II - context analysis"
Proc. FJCC, Vol.33, 1139-1199, 1968

92. Vossler, C.M., & Branston, N.M. : "The use of context for correcting garbled English text"
Proc. 19th Conf. of Assoc. for Comp. Machinery, 1964

* These references are included at the back of the thesis.

PUBLICATIONS

Dynamics of the perception of patterns in random learning nets

M C FAIRHURST and I ALEKSANDER

The Electronics Laboratories, University of Kent, Canterbury

Abstract. This paper presents new results obtained with randomly connected networks of digital learning elements. The networks are dynamic in the sense that feedback connections are not excluded. It is known that after a period of training on two classes of patterns, the network enters few and short cycles of state space activity when exposed to patterns. These cycles distinguish between the input patterns. In this paper it is shown that the net remains in a discriminatory cycling activity even when the incoming patterns are removed. The latter is a remarkable phase in the net's 'perceptual' process which greatly adds to its recognition ability.

1. Introduction

This paper reports on the latest of a series of investigations on the behaviour of randomly connected nets of SLAM (stored-logic adaptive microcircuits).† It has previously been shown (Fairhurst and Aleksander 1971) that a network (connected as described in § 2.1) enters very short cycles of internal states‡ when patterns are *held* at its input. There are fewer internal states patterns than input patterns hence the net performs a natural clustering operation. This is seen as the first phase of a perceptual operation taking place within the net. Our more recent studies have been concerned with the residual state cycling activity in a trained net (the method of training being described in § 2.1) once the *input pattern has been removed.* It will be shown that in the absence of the input pattern the net enters even fewer cycles for each pattern class. This is somewhat remarkable as it implies that the learning net retains a memory of the input pattern which causes it to 'live' in specified state cycles from which it finds it 'easier' to recognize input patterns. This is considered to be a most important second phase in the 'perceptual' process taking place within the net.

2. Experimentation

2.1. Connections and the 'aging' process

In these experiments 36, 3-input SLAMs are used. We label these inputs x_{jk} where j is the jth input of the kth SLAM ($1 \leqslant j \leqslant 3$, $1 \leqslant k \leqslant 36$). The input to the SLAMs is therefore a 3×36 matrix. We call this the X matrix. It is considered that the input to the entire system is a 6×6 matrix to which binary patterns may be applied. We call this the I matrix. Each element of the I matrix

† A functional description of SLAMs is given in the appendix, but it should be noted that they are almost identical to random access memory (RAM) microcircuits.

‡ An internal state is defined as the pattern of 0s and 1s at the output of all the elements.

is connected to three, randomly chosen elements of the X matrix in such a way that no two of the X elements belong to the same SLAM (ie they have different k values in x_{jk}). Also no x_{jk} is connected to more than one element of I. The output of the net forms another 6×6 matrix (the O matrix) whose ordering is arbitrary. There is an additional 6×6 matrix which consists of the 'teach' inputs of the SLAMs. We call this the T matrix. We label the elements of T to correspond with the elements of O. That is, t_{ab} of T corresponds to the SLAM which supplies o_{ab} of O. The feedback connections are made in such a way that elements of O are ORed (after passing through a unit-delay device) with corresponding elements of I. That is, each SLAM input 'sees'

$$i_{ab} + o'_{ab}$$

where o'_{ab} is the output at o_{ab} after a unit time delay. Training includes an 'aging' process, some effects of which have been described elsewhere (Fairhurst and Aleksander 1971). The scheme is summarized here for completeness. This process is designed to reduce the domination of the 'last seen' pattern in a training run.

We are concerned with 2-class learning, there being one prototype pattern in each class. One of these is an H and the other a T as shown in figure 1. The rest

(*a*) (*b*)

Figure 1. Generator for (*a*) T patterns; (*b*) H patterns.

of each class consists of all patterns at a Hamming distance of 2 from these generators. Training consists of selecting a subset of these classes (usually three patterns from each) and feeding each pattern simultaneously to the I and T matrices. Patterns from each class are shown alternately to the net. All the patterns in the training subsets are first shown to the net with all the T elements active. This is the first 'pass'. The patterns are fed to the net for a second pass with only 18 randomly selected teach terminals active, nine of these are active on the third pass, five of those on the fourth pass and two of the five on the fifth pass. This terminates the training.

2.2. *Experimental method*

It is already known (Fairhurst and Aleksander 1971) that an unsupervised learning network with ORed feedback connections (such as the system described in §2.1) is able to perform a clustering operation on two classes of input patterns by means of the aging process introduced during its training period. The effect of the OR function is to maintain a stable pattern in the feedback path corresponding to a region of 'recognition' in the state space of the system (see Fairhurst and Aleksander 1971).

We now consider in more detail a network trained in this way, with particular emphasis on the way in which the recognition properties are retained when a pattern is not forced on the net. To test the net, a pattern (stimulus) is presented to the net and held there until a stable state or recurring cycle of states occurs. The stimulus pattern is then removed (by forcing every input to a logical zero) and the output sequence again observed. Clearly the removal of the input is equivalent to the removal of the OR function and allows the net to undergo unrestricted cycling on *all* its feedback connections. When the output sequence again becomes stable (either as a single stable state or a cycle of states) the original stimulus is replaced at the input and the output sequence again allowed to stabilize.

In all, 16 T patterns and 12 H patterns are tested, these being chosen at random from within each of the two groups.

3. Observed results

In order to draw any conclusions about the effect of *training* the system, it is first necessary to examine the behaviour of an untrained network.

The stores of the SLAM elements are given random logic functions such that, on average, they are half filled with logical 1s. Under these conditions the net has received no information about the two different classes of input on which it is required to operate. The experimental test procedure outlined in § 2.2 is then carried out to discover the state space behaviour of the system.

It is found that the untrained net does not distinguish between the two input classes. The resulting state cycles show no sign of clustering with the input held constant, and on removal of the input, the state space trajectories for both T inputs and H inputs merge into a common dominant recurring cycle of 16 states.

We now consider the effect of training the net on six patterns, three from each class. As described previously (Fairhurst and Aleksander 1971), when the net is tested with a single pattern held constant at the input, the resulting state cycles do cluster, indicating that the system is forming an 'internal' representation of T's and H's. On removing the input pattern (ie reducing the input to all logical 0s) the net starts its state space trajectories from the states reached in the last part of the experiment where the input was held constant. Typical trajectories are shown in figure 2. Clearly, the system now behaves in a manner very different from that of the untrained net. It is found that the system enters one of few cycles, these cycles corresponding to T inputs being *completely distinct* from those corresponding to H inputs as is noted in figure 2. It is interesting to note that for all the H's tested only one cycle was found, this consisting of the single state shown.

The significance of these results lies in the fact that no state space trajectory resulting from an H input enters a T cycle, nor vice versa. The two input groups are now even more clearly separated since only one element of each cycle need be detected. Thus the recognition properties of the net are being retained and indeed enhanced even in the absence of an input.

Finally, we record the effect of replacing an input while the net is in the last found cycle. As shown in figure 3, this forces the system back into the region of state space previously associated with the input pattern.

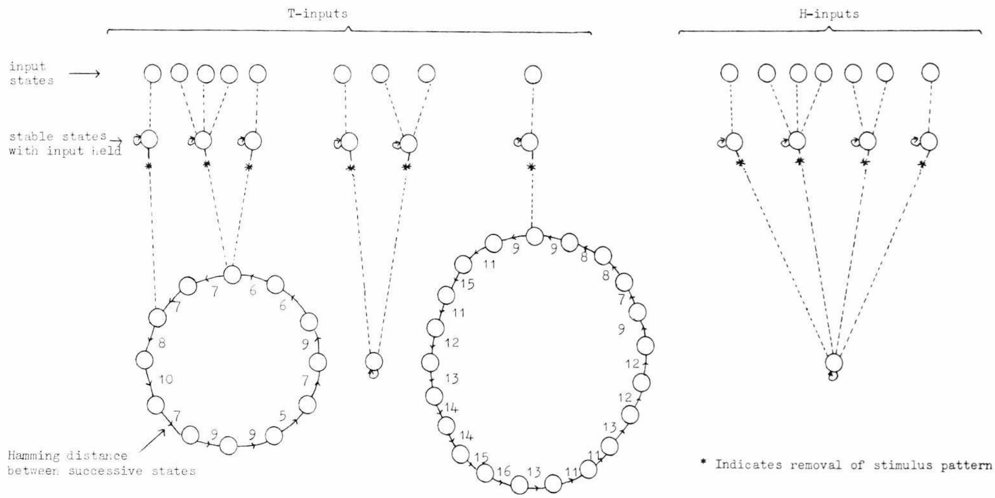


Figure 2. Pattern clustering in the net.

Figure 3(*a*) shows the state space behaviour for a typical T input. With ORed feedback a stable state is reached (in Hamming distance terms, this state will in general be similar to the corresponding generator). On removal of the input the sequence of states eventually reaches a recurring cycle of, in this case, 12 states. Replacing the T input (figure 3(*b*)) drives the system back again into the original stable state. Experimentally this behaviour was found to be independent of which state was chosen as the exit point from the T cycle. If, however, an H input is forced on to the net while in the T cycle (figure 3(*c*)), the system returns to a different region in state space—in this case a cycle of

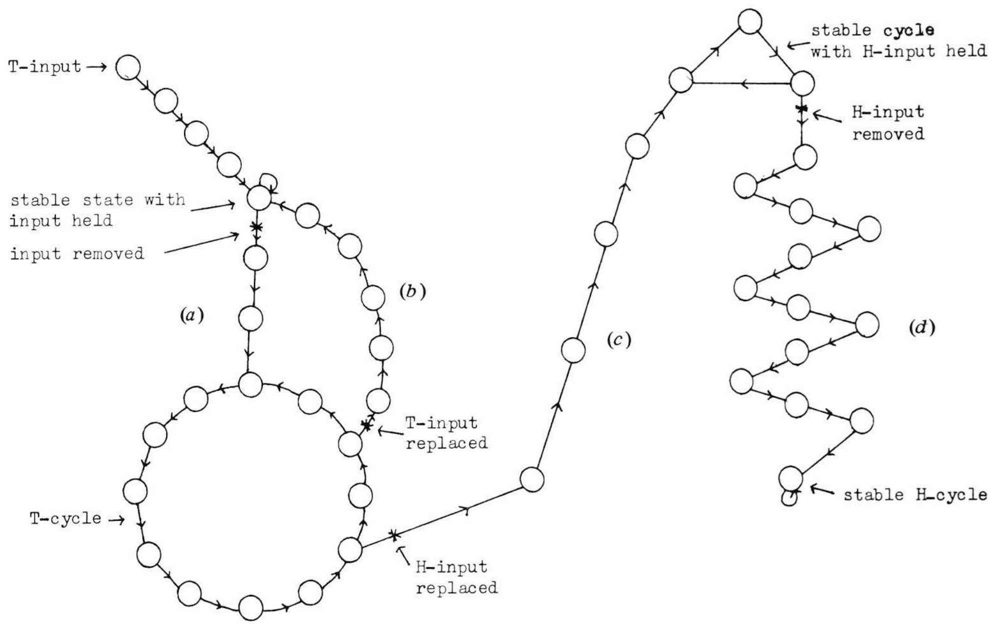


Figure 3. Typical state space dynamics.

three states—which are close in Hamming distance to the H generator. Finally, if this H input is removed (figure 3(*d*)), then the system does *not* fall back into the T cycle but moves to the familiar cycle associated with patterns of the H group.

It is clear, therefore, that the net retains its recognition ability even in the absence of an input stimulus pattern. The pattern at the input simply switches the net into pockets of state space associated with that input during training.

4. Conclusions and analysis

The results presented here show how it is possible to identify two distinct phases in the 'perception' operation of a digital learning net. The first phase corresponds to the 'short term' effect of entering a stable region of state space activity while the net is able to 'see' the stimulus pattern. The second phase consists of the longer term recognition process when even after the removal of the stimulus, state space activity is confined to distinct regions which can be identified with the two possible pattern groups.

The salient features concerning perception in a learning network which this paper seeks to emphasize may therefore be summarized as follows. First, 'perception' by the system can be defined as the entry into specific areas of its state space, these areas being determined by the classes of input patterns to which the net is exposed during training. Second, the existence of few and distinct cycles related to classes of input demonstrates that the input classes are not only remembered by the system even in the absence of a direct stimulus, but also that the active net is forced to 'live' in state space areas related to the environment to which it is exposed during training.

So far, only empirical results have been presented. As pointed out by Kauffman (1969) even untrained nets defy rigorous analysis. However, it is largely as a result of Kauffman's empirical work with such nets that we are now able to *expect* a randomly connected network to act stably. And indeed, the results presented above are an example of a tendency towards stable behaviour in a randomly connected net. In the case of a trained net it is possible to provide a nonrigorous rationale for this behaviour. The most important factor is the net's property to respond in a similar way to patterns that are close in Hamming distance to those seen during training. This property is analysed fully by Aleksander (1970). We call it the S property (S, for similarity).

Treating first the case where a pattern is *held* at the input of the net, we note that owing to the ORed feedback connection only the elements of the I matrix which are at a logical 0 in the input pattern can receive feedback information. Also, because of the method of training, pattern P at the input will produce a pattern P' at the output where the Hamming distance between P and P' is in itself relatively small. Only those binary points that are 1 in P' and 0 in P are transmitted back to the input modifying the input pattern to P''. Now P'' is closer in Hamming distance to P than is P' therefore the likelihood of producing P' again at the output is high owing to the S property. Should this not happen, it is easily seen that the net will go from state to state until either a single state remains at the output or a short cycle is entered. Now, if a slightly modified version of P is held at the input, say P''', the net will enter a new cycle, and the likelihood of this being the same as that entered for P is high, again owing to the S property.

Consider now the case where some terminal cycle has been entered say P_T and the input pattern is removed. P_T will be the only input to the net. This still bears a resemblance to the generator of the class to which P_T belongs. Thus the response to P_T (say P'_T) will resemble P_T but not quite as much as P'' resembled P in the example above. P'_T is fed back in its entirety giving rise to a new response P''_T and so on. The likelihood of reproducing the same pattern at the input is now reduced, but the tendency to generate chains of patterns close in Hamming distance is still retained. Hence the existence of *longer* cycles in this condition. It is precisely this lengthening of cycles which causes the same cycle to 'sweep up' more of the trajectories arising from terminal states (like P_T) which belong to the same class as P_T.

Much work remains to be done on the allowed Hamming distance between generator patterns which retain distinct final cycles in a net of a particular size. From a practical point of view it should be stressed that only one pattern in each cycle need be detected in order to recognize the class of the input pattern, hence the above net has been shown capable of clustering naturally 16 T patterns into 3 patterns and 12 H patterns into only 1 pattern.

Acknowledgments

The authors wish to acknowledge the support of the UK Science Research Council.

Appendix

Functional description of a SLAM: A SLAM element is a random access memory used as a variable-logic/logic element. The memory address terminals $(x_1, x_2, x_3 \ldots)$ are the inputs of the logic element, the single-bit readout terminal o is its output and the single-bit write terminal t is referred to as the 'teach' input. The 'teach clock' (h) input enables the writing mechanism.

As an example, the function of a 2-input element is given by

$$o = \phi_0 \bar{x}_1 \bar{x}_2 + \phi_1 \bar{x}_1 x_2 + \phi_2 x_1 \bar{x}_2 + \phi_3 x_1 x_2$$

where

$$\phi'_0 = \bar{x}_1 \bar{x}_2 th + \bar{h}\phi_0$$

$$\phi'_1 = \bar{x}_1 x_2 th + \bar{h}\phi_1 \ldots$$

(Note that these are sequential equations, where ϕ_j is the 'present' state of a storage element and ϕ'_j the 'next' state.)

References

Aleksander I 1970 *Electronics Lett.* **6** 134
Fairhurst M C and Aleksander I 1971 *Electronics Lett.* 7 24
Kauffman S A 1969 *J. Theor. Biol.* **22** 437

Kybernetes, 1972, Vol. 1, pp. 11–18

AN AUTOMATON WITH BRAIN-LIKE PROPERTIES

I. ALEKSANDER and M. C. FAIRHURST
Electronics Laboratories, University of Kent, Canterbury, England

(Received August 2, 1971)

Starting with a Moore-type automaton the bases for a brain-like sequential machine are laid down. The problem is considered both at the level of a physical structure and a state structure. The logic is cellular and variable to accommodate learning and generalization. It is shown that this structure can "learn to live" in a consistent environment. Concepts such as recognition and recall of environmental events, short-term memory, data generation (analogous to speech production) and attention are shown to be natural attributes of the model.

1. INTRODUCTION

The topic of brain modelling has been tackled by numerous authors, too numerous, in fact, to be cited individually. This is not a review paper and its main aim is to suggest a new approach to the subject.

Central to this approach is the definition of an automaton whose state structure is associated with the brain. Clearly, the brain must be an automaton (in the rigorous sense[2]) of some kind. Closest to this approach is the work of Caianiello[11] in which a model based on threshold adaptive elements with linear, and hence mathematically convenient, properties. In our approach we also use concepts of learning networks[2] but do not restrict ourselves to linear situations. Thus where Caianiello's work is based on classical concepts of linear mathematics, our approach is more general, drawing its formalism from the theory of automata.

Clearly, only the most primitive of brain functions are considered, but an attempt is made to keep the door open for future sophistication.

Specifically, the following concepts are tackled:

a) Adaptation to or recognition of an environment.

b) Recall of environmental events.

c) The production of information (e.g. speech).

In a subsidiary way, this enables us to seek a characterization for phenomena such as short-term memory and attention. The technique adopted throughout, is to lay a foundation with a very general model which is bound to be correct but structurally weak (in a mathematical sense), and to allow for its moulding into a stronger structure by the judicious use of experimental evidence.

2. BASIC STRUCTURE

If the brain is an automaton in the rigorous sense of the word it must be capable of being described by the 5-tuple

$\langle I, Z, Q, \delta, \omega \rangle$

where I is a set of input messages, Z a set of output messages, Q a set of internal states, δ a mapping $Q \times I \xrightarrow{\delta} Q$, and ω a mapping $Q \times I \xrightarrow{\omega} Z$ (or mapping $Q \xrightarrow{\omega} Z$, the decision being made on the basis of whether a Mealy or a Moore model is assumed—see later).

I is a finite, but very large, set. We assume that it consists of *all* the possible combinations of activity or non-activity of the sensory receptors leading to the brain. The nature of this exercise is such as not to compel us to state exactly where "the brain" starts and ends. As a rule, wherever there are neurons or other *processing* (rather than *transmitting*) cells, *that* is considered to be part of the brain.

Z is the set of *overt* responses that can be generated by activity or non-activity of neural pathways. These are considered as being measurable by an external observer. Both in the case of Z and I it is clear that the distinction between continuity and discreteness is a fine one, but we proceed by assuming discreteness hopefully without loss in credibility, and allow that the sets may be large.

It is in Q that the major part of the activity of our model resides. We *must* postulate a set of internal states simply because in no way can we argue that a *combinational* model would suffice.† Much of our

† Even though some Stimulus–Response exponents do precisely that. They assume that the brain is a variable logic combinational system. If this were true it would not be possible for humans to perform any sort of sequential activity (such as singing a song).

modelled brain activity will be seen to consist of trajectories in the state space of Q, again a finite and large set.

The mapping δ determines the nature of the trajectories in Q, and it will be seen that variability of this mapping provides a mechanism for the property of *learning*.

It is well known that the Mealy ($Q \times I \xrightarrow{\omega} Z$) and Moore ($Q \xrightarrow{\omega} Z$) models are equivalent in the sense that one can always be derived from the other, therefore the choice in the model, at first sight, seems arbitrary. A little introspection, however, biases the choice towards the Moore model. The defined outputs Z seems to be directly related to the internal activity in Q. For example, we know that a human subject can keep a speech-signal active even in the face of incoming information such as, say, the ringing of bells. The speech is not corrupted if the bells ring in moderation, and thus Z seems a strong function of Q and a weak one of I. Naturally, for an immoderate amount of noise the speech may be interrupted, but mainly due to distraction or lack of attention. This is due to the influence of the noise in I on the activity in Q rather than its direct effect on Z. Thus a Moore model is assumed from here on.

3. PHYSICAL STRUCTURES, STATE STRUCTURES AND LEARNING

In abstract modelling, particularly in automata models, it is fashionable to leave the model at the abstract level of the 5-tuple discussed in the last section. Here, however, it becomes important to relate the model to a physical structure and that, in particular, the physical structure shall be composed of electronic logic components. In this way the model remains "computable" in the sense that it can be realized as an electronic network. The reason for doing this will become clearer as the model is developed, but it rests mainly on the fact that physical realizability places realistic constraints on the model, hopefully similar to those that exist in the physical realization of the brain.

The physical embodiment of the general 5-tuple model must fall, as is well known, within the framework shown in Figure 1.

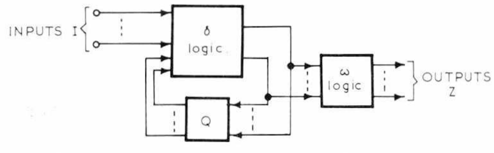


FIGURE 1 General Moore-type model.

It must be stressed as strongly as possible that if Figure 1 is a model of the brain, the element marked "memory Q" is not *the* centre for memory. The box marked Q need only be a set of delay elements, the pattern of signals on which represent an element of the set Q. Binary signals are assumed throughout, thus, if the order of the set Q is N, Q in Figure 1 consists of $\log_2 N$ delay elements. It is now assumed that the delays are locked; that is, the pattern at Q is allowed to change only at discrete intervals of time. This is part of the general quantized nature that has been assumed for the model and is done without loss of generality since its "fineness" is not specified. As implied previously therefore, the element Q is a temporary store for $q(t) \in Q$ which allows the chaining operation:

$$q(t+1) = \delta(q(t), i(t)) \tag{1}$$

where t is a time integer, $q(t+1) \in Q$, and $i(t) \in I$. This chaining operation defines the state-to-state transitions of the system; that is, it determines the *state structure*. Due to the imposition of binary values δ becomes a Boolean logic function, as does ω in the equation

$$z(t) = \omega(q(t), i(t)) \tag{2}$$

where $z(t) \in Z$. Thus, δ and ω in Figure 1 represent collections of logic elements.

So far, all that has been defined is a Huffman[3] circuit model which must hold true for all automata.

Clearly, it is now necessary to restrict this a little by finding a feasible learning mechanism. Clearly, due to its generality, the model as it stands could include a learning mechanism. For example, in the state diagram in Figure 2 we assume that the automa-

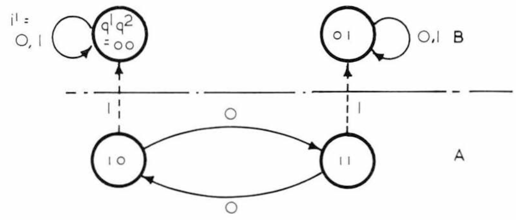


FIGURE 2 Examples of a state structure before learning (A) and after learning (B).

ton "at birth" behaves as indicated by part A of the diagram (the system has one input wire i^1 and two state wires q^1 and q^2, there are no outputs).

Learning consists of shifting to part B of the diagram at the arrival of a "teach" pulse $i^1 = 1$

(where, normally, $i^1 = 0$). This entire procedure may be modelled by Eq. (1):

$$q^1(t+1) = q^1(t).\bar{i}^1(t)$$
$$q^2(t+1) = q^1(t).q^2(t)$$

($. \triangleq$ Boolean AND $\bar{i}$ reads: not i).

We note that q in Eq. (1) represents a vector

$$\begin{bmatrix} q^1 \\ q^2 \end{bmatrix}$$

and that δ above is a fixed vector function. [This type of notation will be used throughout this paper, that is, letters with superscripts are vector elements (e.g. i^1, i^2, z^1, z^n, q^k), whereas small letters with no superscript [such as $q(t)$, $z(t+1)$] are used for the entire vector.]

Exactly the same effect may be achieved by making δ a *variable* logic function which causes the state diagram of the automaton to change from that of A to B. Now only one state variable q^1 is required (say, $q^1 = 0$ for the left-hand state and $q^1 = 1$ for the right-hand state) there is no direct input, thus the labels on the arrows disappear.

For A the δ function is:

$$q^1(t+1) = \overline{q^1}(t)$$

whereas for B, the δ function is

$$q^1(t+1) = q^1(t)$$

Thus if we postulate the existence of an element which performs the function

$$q^k(t+1) = \phi_i \overline{q^1}(t) + \phi_2 . q^1(t) \tag{3}$$

($+ \triangleq$ Boolean OR). A is achieved by letting $\phi_1 = 1$ and $\phi_2 = 0$, while B is achieved by letting $\phi_2 = 1$ and $\phi_1 = 0$. Exactly how these variables are set will be tackled later, here it is merely established that *learning* in our model resides in the *variability* of the δ logic (and indeed the ω logic) of the model. We have thus arrived at an automaton as in Figure 1, but with a variable state structure and a variable output mapping. Learning being defined as the variability in this structure fits in rather well with a common definition of "learning" as a *change in behaviour*.[3] Indeed, the state structure of an automaton *is* its behaviour.

4. MICROSCOPIC LOGIC STRUCTURE

Two overriding principles guide us in modelling our logic structure:

a) the structure shall be cellular,

b) no specific order should be assumed in the interconnection of the cells.

These principles are derived from a need to impose similar functional restrictions on the model, as are likely to be in force in the brain itself. Thus if the structure is to be cellular and non-ordered, attention must be given to the function of each cell and the way these are connected.

Any logic cell may be described by three parameters

$$\langle m, n, F \rangle$$

where m is the number of inputs, n the number of outputs, and F the set of functions relating the inputs to the outputs.

It is assumed that $n = 1$ for all our cells. This can be done without loss of generality, as it is easily shown that an n-output cell is equivalent to n, 1-output cells. We do not need to specify m too closely, except that for the sake of physical "computability" it could be quite low, perhaps between 2 and 5. Indeed, it has been shown[4] that random networks of cells with $m = 2$ have a relatively sophisticated "intelligent" behaviour. Given this situation, one allows F to be the set of *all* of the 2^{2^m} logic functions of m inputs.

At this point it becomes necessary to comment on the comparison between the cells defined above, real neurons and neuron models. At first sight, it appears that in the quest for physical computability, the restriction, in terms of the number of inputs imposed on the basic cell of the model, is greater than that imposed by nature on the neuron—the basic cell of the brain. However, this is not so.

It has been assumed in decades of neuron modelling,[5] that the neuron performs only linearly separable functions of its inputs. This means that for a neuron with m inputs, the order of F is *very much less* than 2^{2^m} for large m. Now, this is precisely the case with an *assembly* of our basic logic cells. For example, the network in Figure 3a performs a number of functions, which is of the same order as the

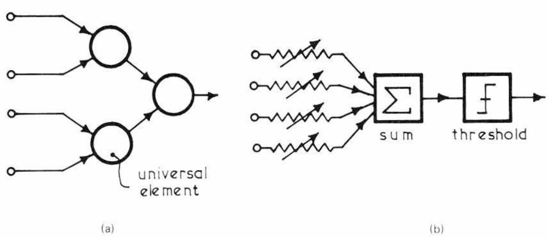


FIGURE 3 Examples of non-universal logic circuits.
a) Composed of universal elements.
b) A linear separator.

neuron model in Figure 3b. There is also a similarity between the sets F of the two models.[10] Thus it is possible to assert that even though our basic cell is clearly different from a neuron, this does not preclude its use in a large assembly of such cells which may well behave in a manner analogous to an assembly of neurons. It is one of the salient contentions of this paper that an assembly of variable-logic cells behaves in a fashion analogous to that of the brain, and that the precise functions of the cell elements are not very important. In other words it is possible to explain phenomena such as adaptation to the environment, recall, etc. by referring only to the bulk properties of a cell assembly.

This leads directly to the question of the way in which the cells are interconnected. The guiding principle that they should not be connected in a systematic way has already been stated. The reason for insisting on this is that there *is* no source of an algorithm whereby this network might be connected. Therefore if any particular network of cells can be shown to be useful as a model it should be one with relatively random connections. In nature, physical constraints may well impose some kind of order on the connections between cells. For example, only neighbouring groups of cells may be heavily interconnected, whereas distant cells may not be. Indeed, the advocacy of this paper is that the actual interconnections between cells are of secondary importance.

It is a simple matter to relate the concepts of a cellular structure to the Moore model of Figure 1. This is illustrated by the "random" net in Figure 4(a), where some cells with inputs to the environment and some with outputs to it are shown. All are "randomly" interconnected to cells with no inputs or outputs. Clearly, this network may be re-drawn as in Figure 4(b), where the relation to the Moore model becomes evident.

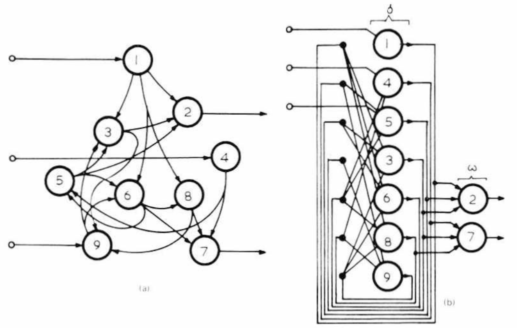


FIGURE 4 Equivalence between network topologies.
a) A random net.
b) Re-drawn in the Moore form.

5. WHAT IS LEARNT, AND HOW?

Learning in the model has already been defined as a variation of the δ and ω functions. Referring now to the cellular structure, this implies changes in the particular functions (from F) which are performed at any time. In fact the overall operation of each cell must assume a form similar to that of Eq. 3. For example, for $m = 3$, the overall operation of a cell may be expressed as

$$f = \phi_0\, \bar{a}\bar{b}\bar{c} + \phi_1\, \bar{a}\bar{b}c + \phi_2\, \bar{a}b\bar{c} + \phi_3\, \bar{a}bc + \phi_4\, a\bar{b}\bar{c} + \phi_5\, a\bar{b}c + \phi_6\, ab\bar{c} + \phi_7\, abc \tag{4}$$

where a, b and c are the cell inputs, f the output, and ϕ_i a binary variable. This implies that each cell contains a store of 2^m bits, representing the variables ϕ_0 to ϕ_{2^m-1}. The value of these bits uniquely defines the function of the cell, and the set of functions F is seen to be in one-to-one, correspondence to the 2^{2^m} possible values of the vector

$$\boldsymbol{\Phi} = \begin{bmatrix} \phi_0 \\ \phi_1 \\ \phi_{2^{m-1}} \end{bmatrix}.$$

Changes in this vector imply changes in the state structure of the network.

There are many ways in which the ϕ vector of a cell may be changed. It has often been argued that the most time-wasting, and hence unlikely, are those in which a "punishment" signal is transmitted to the cell.[6] Excluding such an arrangement (mentioning it only for the sake of completeness) it appears that a useful way of achieving the change in the vector is by causing the cells to receive information (through a "teach" terminal φ) regarding a *desired* output. To be precise, this means that a cell with some binary pattern on inputs, a, b, c ..., on receiving a 0 or 1 signal at φ will reproduce the signal as its output whenever the same pattern is re-applied to the inputs. This means that in the 3-input device described by Eq. 4, and ϕ_j single-bit storage elements are *addressed* by the pattern on a, b, and c, and then *set* to the value currently at φ. The physical details of such an element may be found elsewhere.[1] These cells are not only physically realizable, but are commercially available as standard electronic microcircuit components called SLAMs (Stored Logic Adaptive Microcircuits).

It is important to decide on the rôle that the additional terminals φ play in the overall model. Clearly, the brain has no specific set of "teach" channels connecting it to the environment. In fact, *no matter how the elements of the brain change their* *logic function*, the "teach" information *must* come from the five senses. Equally, therefore, it is specified that in the automaton model the φ terminals be made to *sample* the same information that is being fed to the net inputs. Rather than being an arbitrary assumption, it is a contention of this paper that this arrangement is responsible for the adaptation of the model to the environment and its subsequent "intelligent" behaviour. This point is pursued in subsequent sections, whereas one specific qualification must be added here.

Clearly *something* must decide whether the elements are in a learning mode or not. Here it is simply stated that an aging process is taking place and that the probability of changing the stored vector in each cell decreases (say exponentially) with time. The time constants of such changes have a broad distribution, but the maximum probability of a change in each cell is "at birth." This is a slight departure from the rather simple physical realizability of the model so far. Indeed, as a future refinement of the model, a probabilistic behaviour will have to be built into the cells. This is easily realized by the introduction of noise into an otherwise deterministic system. Except where otherwise stated, it is not necessary to evoke such a probabilistic behaviour in the cells—the principles discussed in the rest of this paper do not depend on it.

6. RECOGNITION OF THE ENVIRONMENT

At this stage we refer to the block diagram in Figure 5. This differs from Figure 1 in the sense that the φ

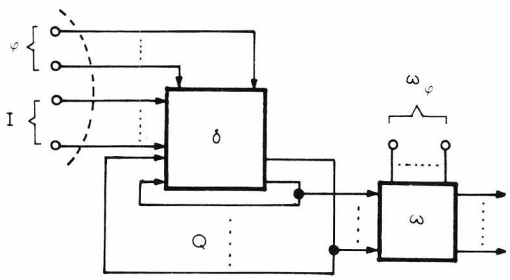


FIGURE 5 A Moore model with "teach" inputs.

connections are shown. For the time being, the precise accommodation of the φ terminals of the ω logic is left undefined, as, initially, only the δ logic is of concern. It is assumed that the system is "born" as some specific time at which it becomes exposed to the environment for the first time. That is, its "senses" become active and both the teach terminals φ and the inputs begin to receive information.

Consider first the frequent repeats of presentation of a pattern M (say the face of the mother) among a bewildering bombardment of other inconsistent patterns. Let us say that the part of M reaching the input terminals is M_i and that reaching the "teach" terminals is M_φ. Since the teach terminals are largely active, M_φ becomes an output of the δ-logic, associated with M_i. That is, the δ-net will *respond* with M_φ at the arrival of M_i at the input. The effect of the feedback loops is such that M_φ will reach the input terminals and a compound pattern $(M_i + M_\varphi)$ will also be associated with an output M_φ. The resulting effect in state space is shown in Figure 6,

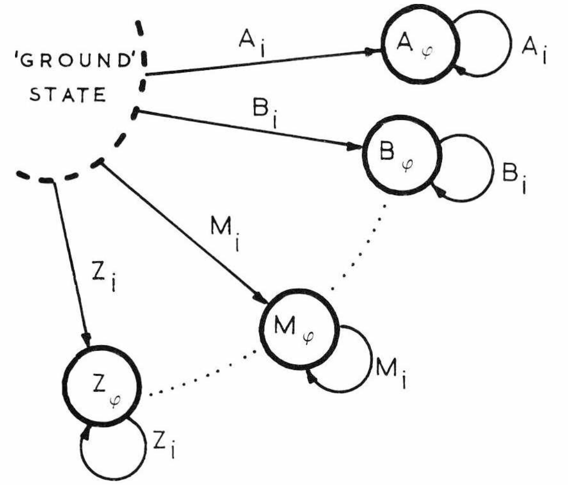


FIGURE 6 The recognition of single patterns from a ground state.

where the result or exposure to other patterns $(A, B \dots Z)$ is also indicated. The "ground state' has been included only to illustrate the concept of the formation of the "stable" cycles like M_φ. However, during "consciousness", if there is a continual change between the stationary input patterns, the state space would develop as in Figure 7. Here, K signifies a return to some ground state

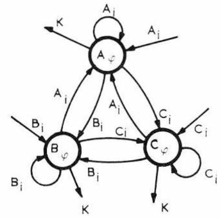


FIGURE 7 Alternative recognition structure with optional returns to a ground state.

(when the input is not in the class of consistently appearing images). The automaton thus learns to "live" in a state space which corresponds to consistently recurring input images, and, in due course, the ground state would disappear. This principle applies not only to *single* images but also to consistent image sequences.

If, for example, input patterns P, Q and R *always* appear in *that* sequence, the state transitions will form as shown in Figure 8(a). This triplet of states is hardly different to a single state and could be represented as shown in Figure 8(b).

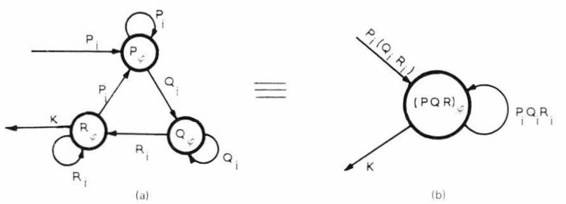


FIGURE 8 Pattern sequences that may be considered as single patterns.

Thus, it is possible to define *recognition* of an environmental pattern as entry into a state space region which corresponds to that pattern and which is formed during learning by simple exposure of both the φ and I inputs to the pattern itself. Little has been said here about generalization in the system, however the mapping of inputs into stable states is clearly a many-to-one affair as is described elsewhere for simple digital nets.[1, 7]

7. RECALL AND MEMORY

To define recall and memory, it is necessary to assume that a state of inputs exists which does not evoke a direct, recognized response. This is like an "eyes closed" situation in which input signals are effectively cut-off. We call this input O_i. Consider a likely sequence of events if the O_i signal occurs while the system is in the stable state M_φ. Note that the δ feedback maintains the M_φ part of the $(M_\varphi + M_i)$ input image. This means that the state will change to a new state M_φ^1, but due to the generalization of the net there will be a similarity between M_φ and M_φ^1 in the Hamming-distance sense. This gives rise to entry into a new state M_φ^2 and so on until the system enters a *cycle* of states (i.e. one of the elements of the chain is repeated). Typically this cycle consists of only one state as shown in Figure 9. It has been shown elsewhere[1, 8] that the nature of this state trajectory depends on the amount of training

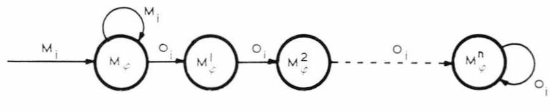


FIGURE 9 Short-term memory: O_i is an "eyes closed" input resulting in a state drift as shown.

previously applied to the system. Briefly, this means that if a system has "seen" many similar versions of M (implying that there are many stable states similar to M_φ), then the chain of states to M_φ^n is short and the state(s) in the final cycle is (are) similar to M_φ. This final cycle is defined as the *recalled* version of M. There are many factors that give this definition a satisfying distinction. Firstly, we know introspectively that the memory of an image has less definition than the original. Secondly, the chaining effect provides a useful mechanism for short-term memory. It has been seen experimentally[1] that if the system has only had a brief glimpse of an image, the chain is long, the states decrease in similarity to M_φ (memory fading) and the final state need bear no resemblance to M_φ at all.

The dynamic nature of the proposed memory mechanism is stressed here. It is very different to the computer-like store models assumed by many brain modellers.[9] It is also stressed that much more experimental work remains to be done on large networks in order to ascertain the precise nature of the state-space trajectories in the above recall mode. Indeed, a theory of such behaviour is the subject of much current research.

8. EFFECT OF LONG FEEDBACK LOOPS (SPEECH PRODUCTION)

As is probably the case in the human brain, the existence of a long feedback loop between the overall output and the input of the model plays an important part in the development of its "intelligence". It will be shown here that through the action of this loop the model can learn to associate an input image with an appropriate output message. For clarity we shall use the analogy of the automaton's learning to "say" "mah-mah" on seeing its mother's face.

Let M be the input pattern (of the mother's face). This gives rise to the usual stability of the δ logic in state M_φ. Now assume that consistently and simultaneously with M the sounds "mah-mah" (symbolically, just *m-a-h*) are applied to the automaton. The resulting state activity is shown in Figure 10. Again,

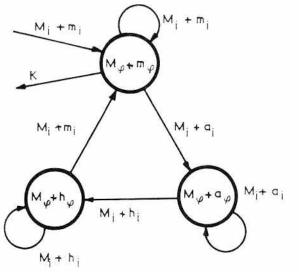


FIGURE 10 Recognition of the *m-a-h* sound sequence (entry for M not shown).

due to the generalization of the δ net, it is possible for the system to enter this cycle of activity for *either* only the input M *or* only the input *m-a-h*. Here one must remain aware of the fact that it is not a real physical situation (i.e. a baby) that is being described, but an over-simplified one with hopefully analogous characteristics.

It is assumed at this atage, that the ω logic is untrained (i.e. set at random). It is also assumed that the output repertoire of ω is not numerous. That is, ω maps in a many-to-very few way from the internal state to the output (i.e. there are only a few output connections from ω). Thus, during any internal activity of the system the output will be emitting random messages from its repertoire. Assume that *m-a-h* or something similar to it is part of the repertoire and that it is generated from time to time. Each time that *m-a-h* is produced it will be *recognized* in the sense that the δ logic will enter the appropriate cycle.

Now it is necessary to specify the connection of the "teach" terminals of the ω logic. Following our previous philosophy, the teach terminals of ω (ω_φ) must sense some part of the input. But, due to the way that these terminals affect the eventual output of the net,† it is assumed that the ω_φ terminals are restricted to the sensing of the "*muscular*" effort involved producing the output. That is, the ω_φ terminals sense the speech *production* effort needed to generate *m-a-h*. Call this m'_φ-a'_φ-h'_φ. As a result of the recognition in δ, there would be a *consistent* and, for all intents and purposes, simultaneous stimulation of m'_φ-a'_φ-h'_φ at the "teach" terminals of ω and the m_φ-a_φ-h_φ at the inputs of ω. The association is formed with the result that when *m-a-h* is "heard" by the automaton it enters its m_φ-a_φ-h_φ cycle in δ and generates m'_φ-a'_φ-h'_φ (as speech) at the output of ω. This would also occur for a *visual* input M. That is, the automaton has learned to "speak" in response to the recognition of a visual or aural stimulation.

† A pattern P at φ for an input K at inputs I will subsequently appear as P at the *output* when K is at the input.

Some comment is required on this state of affairs. Firstly, the above mechanism seems too rigid in the sense that we know that babies do not *automatically* say "mah-mah" when they see their mother's face. The element of *choice* as to whether to respond or not seems to be missing. It is thought that this is an example of "attention" which will be touched on in the next section of this paper.

Secondly, the assumption that the ω terminals are associated solely with the muscles of speech production implies that an efficient model might have many, somewhat disjoint output logic systems such as ω, one for each significant muscle group (e.g. head-turning muscles, arms, hands, etc.). It is not known to the authors whether there is physical evidence for this in animal brains. Finally, a prerequisite which allows the model to learn to "speak" is that it must build up internal state cycles on information similar to its own output repertoire. In humans this would mean that a baby learns to speak because it is exposed to "baby talk". It is not known to the authors whether examples exist where babies have not learned to speak through *not* being exposed to enough "baby talk".

9. ATTENTION

This is probably the most baffling of all brain activities as far as the brain modeller is concerned. One of the difficulties is that the concept covers many different activities and it is by no means certain that all have the same mechanisms. These activities may be broadly summarized as

i) the reception and recognition of one set of data among others (the cocktail party problem);

ii) the choice of muscular action (whether to walk left or right or whether to turn one's head or not);

iii) the choice of thought (it is clearly possible to control one's internal state activity).

The first of the above effects may be explained by the fact that internal state trajectories *must* take place along meaningful (i.e. previously learnt) paths. This is a corollary to the argument put by many workers that the brain anticipates the high transition probabilities of sequences of messages.

To explain the other forms of attention, it may be wrong to represent "thought" as a single coherent state activity. That is, it is necessary to assume that the loops may be partitioned into groups within each of which some form of sub-activity is taking place. Attention is the result of the interactions between hierarchies of such activity. Consider the following example.

An automaton model is watching a programme on television (activity T). Some of its feedback loops are engaged in the recognition of this information. Someone walks into the room and starts saying something (activity S). Through introspection and experimental evidence we know that a human subject will not be able to attend fully to both, but, while paying full attention to one he can still devote a *little* attention to the other. To explain *any* co-existence between these two activities one postulates the existence of a second set of feedback loops which enters an internal activity corresponding to S. If in each loop the next state were dependent solely on the present state in *that* loop, the two activities could co-exist side by side. This is not the case, implying that there is an interaction between the loops in the δ logic. Also, due to the assumed connections of the teach terminals this interference is very likely. To make "sense" of the situation the automaton must get out of this confusion. One way of doing this is by muscular action. It can turn its head (say) and look at S rather than T, and then return to T. This is as much a *learnt* response of the ω logic as the speech production discussed in the last section and provides a modelling basis for the type of attention mentioned in (ii) above.

We see that in addition to the two loops involved in the straight recognition of incoming data, a third loop (the long-routed one in the above case) is responsible for the control of attention. Exactly the same mechanism may be postulated for attention of type (iii), except that in this case the third controlling loop would be a short-routed internal one, that is another loop around the δ logic. Although much work remains to be done on the way in which these attention mechanisms are learnt, in principle, it has been argued that attention does not fall outside the range of action of the proposed model.

10. CONCLUSIONS

This paper has set out to built up a general automaton-like framework for brain modelling. The following features have been derived:

i) A Moore structure is preferred.

ii) The logic of this structure is cellular and randomly connected.

iii) Learning is a result of associative action within the cells, which only requires a sampling of the environment. No special "teach" signals are assumed.

iv) Long-routed loops play a vital role in the behaviour of the model.

v) The basis for the existence of psychological properties such as, recall, short-term memory and attention has been established, but much work remains to be done on the quantification of these concepts.

REFERENCES

1. I. Aleksander, "Some psychological properties of digital learning nets" *J. Man-Machine Studies*, **2**, 189–212, (April 1970).
2. M. A. Arbib, *Theories of Abstract Automata* (Prentice Hall, Englewood Cliffs, 1969).
3. W. H. Thrope, "The concepts of learning and their relation to those of instinct" *SEB Symposia, IV Physiological mechanisms in animal Behaviour* (Cambridge, 1950).
4. S. A. Kauffman, "Metabolic stability and epigenesis in randomly constructed genetic nets" *J. Theoret. Biol.* **22** 437–467 (1969).
5. W. S. McCulloch and W. Pitts, "A logical calculus of the ideas imminent in neutral nets" *Bull, Math. Biophys.* **5**, 115 (1943).
6. I. Aleksander and R. C. Albrow, "Adaptive logic circuits" *Computer J.* **11**, 65–71 (1968).
7. I. Aleksander, "Microcircuit learning nets: Hamming distance behaviour" *Electronics Letters* **6**, 134 (1970).
8. I. Aleksander and E. H. Mamdani "Microcircuit learning nets: improved recognition by means of pattern feedback" *Electronics* Letters **4**, 425 (1968).
9. D. A. Norman, *Memory and Attention* (Wiley & Sons, New York, 1969).
10. R. C. Albrow, "Multi function logic circuits" *Ph.D. Thesis* (Univ. of London, 1970).
11. Caianiello, E. R., "Outline of a theory of thought processes and thinking machines" *J. Theoret. Biol.* **2**, 204–235 (1961).

Reprinted from *Electronics Letters,* Vol.6, No.16, 6th August 1970

PATTERN LEARNING IN HUMANS AND ELECTRONIC LEARNING NETS

Indexing terms: Brain models, Learning systems, Pattern recognition

The letter reports on a previously unseen similarity between an associative memory model recently proposed for the brain and some electronic systems that have been developed for practical learning machines. This similarity is analysed, and its implications are discussed. Results of a comparison between the learning behaviour of an electronic network and a group of human subjects in a pattern-recognition task are given to complete the comparative study.

The associative memory model (a.m.), proposed by Willshaw *et al.*[1] may be described by the expression

$$a_k = 1 \quad \text{if} \quad \sum_j \phi_j b_j \geqslant T \qquad (1)$$

and

$$a_k = 0 \quad \text{if} \quad \sum_j \phi_j b_j < T$$

where a_k are the N_A binary outputs and b_j are the N_B binary inputs of the associative net; T is some threshold integer and ϕ_j is a binary constant which is set to 0 or 1 during the training period of the net. This is a slightly degenerate form of the classical threshold model of neurons such as is used by Widrow and Hoff[4] and Taylor,[5] which may be described by the same expressions as in eqn. 1 with the difference that the ϕ_j are continuous 'synaptic-weight' constants. The a.m. model is therefore a binary-weight version of the classical analogue-weight model.

Also, the a.m. model is a particular version of a single-layer binary learning (s.l.b.l.) net[2, 3] shown in Fig. 1. Here it is seen

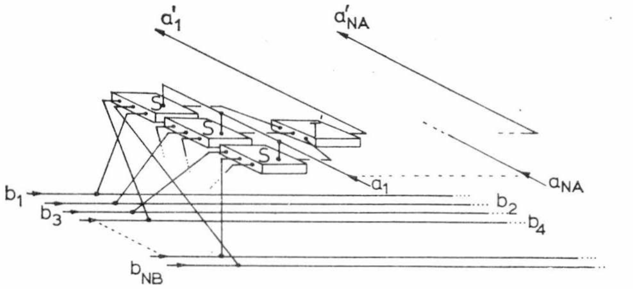


Fig. 1 *Single-layer binary learning-net model*

that the b inputs are sampled as n-tuples by S units (called stored-logic adaptive modules or s.l.a.m.s) each of which is capable of forming a complete association between n inputs and one output. Consequently the S unit contains 2^n bits of storage. The association is formed according to the binary value of the input a_k, a_k' being the output of a threshold decision unit T' such that $a_k' = 1$ if T' or more of the S units are active. It is evident that the a.m. model is equivalent to an s.l.b.l. net with $n = 1$.

Willshaw *et al.* consider the case where the net associates patterns with M_A out of N_A and M_B out of N_B 'on' states and show that, for maximum storage of information and least association error $T = M_B = \log_2 N_A$. In the Widrow model, there is no such assumption, and the adaptation of the threshold is as much a part of the learning process as the adjustment of the synaptic weights. Indeed, one is led to question the meaning of the information-capacity analysis of the a.m. model, since it implies a constant M_A and M_B. It is more likely that nervous tissue has to cope with the association of patterns in which M_A and M_B are not constant. For s.l.b.l. models such as in Fig. 1, assuming no restriction on M_A and M_B, the total amount of information storage implied by a perfect 'A from B' recall is $N_A 2^{N_B}$ bits. Such a model has $n = N_B$ and does not possess the important property of generalisation (i.e. the property of coping with patterns similar to those in the training set). This property arises as a result of making n less than N_B in which case the actual storage of an s.l.b.l. net is $N_A(N_B/n)\,2^n$. The precise nature of this generalisation has been discussed elsewhere.[8] It is well known both experimentally[6] and theoretically[7] that putting $n = 1$ as in the a.m. model results in the least storage and also in the highest error due to overgeneralisation (even with the threshold set at N_B). It is unlikely that a living tissue would evolve into such an unfavourable situation, and it is probable that the Widrow model is physically a closer representation. Nevertheless, we have found that the s.l.b.l. model, besides being easily realised as electronic hardware, has a learning characteristic that may be compared to that of humans in a simple pattern-recognition task.

It is possible to calculate the most probable behaviour of s.l.b.l. systems if the Hamming distances* of a test pattern to those in a training set are known.[8] On this basis, an experiment was carried out to fit an s.l.b.l. model to the results obtained from a pattern learning test on 16 human subjects.

The experimental patterns were derived from a generating pattern shown in Fig. 2*a*. They were formed by randomly changing the squares on this pattern, the number of squares changed being the Hamming distance H from the generator.

Patterns selected as the training set were each at a Hamming distance of 2 from the generator, and thus at maximum Hamming distance of 4 from each other. In all, there were five different sets of patterns, classified in terms of the value of H with respect to the generator pattern. The five values of H used were 2 (the training set), 6, 10, 22 and 28.

The patterns were projected onto a screen. In each of 10 tests, the first five were training patterns ($H = 2$). These were followed by five test patterns selected from the other sets, for each of which the subjects were asked to assess its similarity to the patterns in the training set by means of a 'score' between 0 and 10. 10 signifies a high similarity and 0 a high dissimilarity. The test patterns in each test consisted of one pattern from each set. On average, high responses were obtained for similar patterns and low ones only for dissimilar patterns. The ability to separate these improved with training. Typical distributions of responses to patterns with $H = 2$ and $H = 22$ are shown in Fig. 2*b* as a function of training. The results for other Hamming distances follow a similar form and will be published in greater detail in a longer paper.

Fig. 2*c* shows a similar calculated distribution of responses for s.l.b.l. nets with $N_A = 1$, $N_B = 36$ and $n = 4$. This was

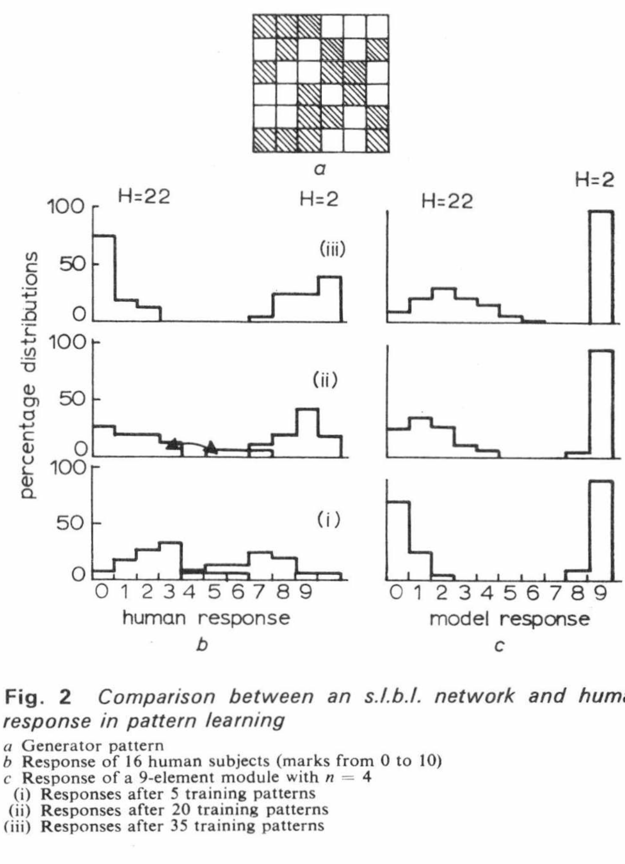


Fig. 2 *Comparison between an s.l.b.l. network and human response in pattern learning*

a Generator pattern
b Response of 16 human subjects (marks from 0 to 10)
c Response of a 9-element module with $n = 4$
(i) Responses after 5 training patterns
(ii) Responses after 20 training patterns
(iii) Responses after 35 training patterns

* The number of binary points by which two patterns differ

obtained by selecting a value of n which gave a reasonable comparison.

The results shown have been calculated according to the principles set out in a previous letter.[8] The calculation is based on a consideration of all the ways in which the network can be connected and represents the most likely result for any one randomly chosen connection.†

It is noted that, where the net is able to distinguish clearly between the two sets of patterns after 'seeing' only five training patterns, the human response shows a certain amount of 'confusion' (i.e. overlapping distribution), although this improves as the number of training patterns increases. The electronic net loses its ability to separate the patterns if it is trained to excess. Human response on the other hand, continues to improve with training owing to its ability to 'forget' infrequently occurring events. In summary, we find it interesting that the electronic system with a capacity of only 144 binary units of information learns in a way comparable to that of humans, after being trained on only five or so patterns, where the humans require more than twenty.

Willshaw *et al.*[1] and van Heerden[9] have commented on the discrepancy between the flexibility with which humans recognise patterns and the apparent rigidity of holographic (van Heerden) and associative (Willshaw) models. Indeed, one interesting side result of the above pattern-learning experiment is that the humans were asked at the end of the experiment to draw a pattern which they thought was similar to the generator. Most of the drawn patterns were within 10 binary points of the generating pattern and two were identical to it.

A mechanism that could account for this type of memory is feedback between the a' outputs and the b inputs of the system. If the a and b inputs were to sample the same incoming information, the net would learn to respond to the 'familiar' pattern by producing a version of the training pattern at the a' outputs. Feedback would cause the net to oscillate maintaining the training pattern at the a' outputs.

This gives a new meaning to the concept of recognition, as it implies that recognised information sets the net into an oscillation which internally produces signals that correlate with the incoming information. The human ability of drawing the generating pattern is analogous to the generation of a self-sustained and self-generated oscillation. Some mechanisms of such oscillations and the way they extend the behaviour of s.l.b.l. nets to the recall and control of sequences have been described elsewhere.[3] This still requires a great deal of research attention and is a concept which may be applied equally to all combinational learning net models including associative and holographic nets.

† There are $\{N_B!/n!(N_B-n)!\} = 58\,905$ ways of connecting the network

I. ALEKSANDER
M. C. FAIRHURST

Electronics Laboratories
University of Kent at Canterbury
Canterbury, Kent, England

3rd July 1970

References

1 WILLSHAW, D. J., BUNEMAN, O. P., and LONGUET-HIGGINS, H. C.: 'Non-holographic associative memory', *Nature*, 1969, **222**, p. 960
2 ALEKSANDER, I., and ALBROW, R. C.: 'Pattern recognition with adaptive logic elements', *IEE Conf. Publ.* 42, 1968, p. 1
3 ALEKSANDER, I.: 'Some psychological properties of digital learning nets', *J. Man-Mach. Stud.*, 1970, **2**, p. 189
4 WIDROW, B., and HOFF, M. E.: 'Adaptive switching circuits'. IRE WESCON Convention Record, 1961
5 TAYLOR, W. K.: 'Pattern recognition by means of automatic analogue apparatus', *Proc. IEE*, 1959, **106B**, pp. 198–209
6 ULLMANN, J. R.: 'Experiments with the *n*-tuple method of pattern recognition', *IEEE Trans.*, 1969, **C–18**, p. 1135
7 HUGHES, G. F.: 'On the mean accuracy of statistical pattern recognisers', *ibid.*, 1968, **IT–14**, p. 55
8 ALEKSANDER, I.: 'Microcircuit learning nets: Hamming-distance behaviour', *Electron. Lett.*, 1970, **6**, pp. 134–136
9 VAN HEERDEN, P. J.: 'Models for the brain', *Nature*, 1970, **225**, p. 177

NATURAL PATTERN CLUSTERING IN DIGITAL LEARNING NETS

Indexing terms: Pattern recognition, Artificial intelligence, Learning systems

This letter reports the results of experiments in which an unsupervised net is found to perform a natural-clustering operation. An 'aging' process is introduced during learning, and the net is randomly connected with outputs feeding back to inputs.

In the field of pattern recognition, much effort has been devoted to the development of clustering algorithms in which measurements taken on patterns of various classes become more closely related to one another within a class after being operated on by the algorithm.[3] We have previously reported that digital learning networks of s.l.a.m. (stored logic adaptive microcircuit)* elements with random feedback connections could be *trained* to enter stable cycles of patterns on their feedback loops as a response to unknown input patterns.[1,2] In our previous work, we carefully selected the training patterns which were fed to the 'teach' terminals of the s.l.a.m.s to achieve this effect, and the scheme relied heavily on the presence of an 'intelligent' teacher. In this letter, we describe experiments which show that a net whose inputs *and* 'teach' terminals sample the incoming patterns during a training period performs a natural clustering operation. An 'aging' process is incorporated in the system. This gives a gradual exit from the 'learning' period which counteracts the deterministic storage characteristics of the s.l.a.m.s, and results in a scheme for adaptation with a less direct intervention from a teacher.

To achieve this aging, it is assumed that 'training' information is clocked into the s.l.a.m. elements for every input pattern, but one allows the network to become gradually less sensitive to information at its teach terminals by removing the 'teach clock' signal from an ever increasing number of elements. As training progresses, therefore, an increasing number of elements in the net will cease to change their logic function in response to patterns at the input.

The feedback path from output to input enables the net to cycle in response to a stimulus pattern, and the pattern in the feedback path is 'ORed' with the original input before subsequent passes through the net, as assumed in other studies. This is one of the many ways of connecting the feedback loops, as discussed elsewhere.[2] The network was made up of 36 s.l.a.m.-8 (3-input) elements randomly connected to a 6×6 matrix at input and output. The state of the system is, in fact, the output pattern, which, at the arrival of a clock pulse, is fed back to the input. This arrangement is shown in block-diagram form in Fig. 1, where the random connections are made between the output of the OR gates and the s.l.a.m. inputs so that each OR gate is connected to three different s.l.a.m.s, and no s.l.a.m. input is connected to more than one OR gate. The 'teach' terminals also sample the incoming patterns, as shown.

2-class learning is considered. For simplicity, an archetype 'T' and an archetype 'H' are chosen as the generating patterns for the two groups, each of which consists of all possible patterns at a Hamming distance of 2 from the respective generator.

The training set consists of two patterns from each of the two groups, and two patterns at a larger Hamming distance (= 10), one from each generator. The net was shown this same sequence of patterns five times, with the number of active 'teach' terminals shown in Table 1.

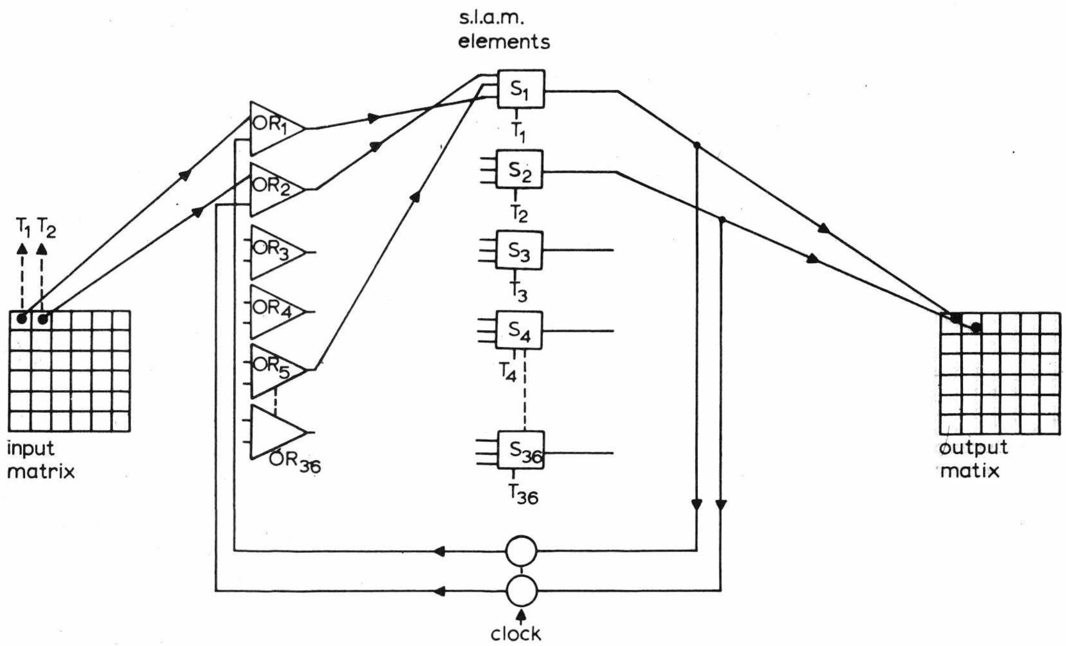


Fig. 1 *Block diagram of system*

Table 1

Pass	Number of active terminals
1	36
2	18
3	9
4	5
5	2

Each set of terminals is a subset of its predecessor. The same sets of 'teach' terminals are cut off for all experiments. The net is always allowed to cycle (with an observed maximum of four transitions) until a stable output is observed.

In this letter, results obtained with a few patterns are presented graphically in Fig. 2. The Hamming distances are measured with respect to the most frequently occurring final state for each group, these being labelled T and H as indicated. It should be noted that only initial and final states are shown, even though the trajectories in state space have passed through intermediate states.

Despite the fact that it is impossible to draw a 2-dimensional diagram in which all the Hamming distances are to scale, clear evidence of clustering is shown in Fig. 2. First,

* A brief functional description of the s.l.a.m. element is given at the end of the letter

one notices that a number of inputs map into the same final state (Figs. 2a and c). Such final states are a property of the net, i.e. they are archetypes generated by the net itself as a function of its training and random interconnections. Typical states are found to be close, in Hamming distance, to the respective generators. Secondly, there is a definite reduction in average Hamming distance between the domain of the input patterns and the range of the final states. Quantitatively, the average Hamming distance goes from 3·5 between the input patterns T to 2·3 between the final states and from 3·6 to 2·5 for H. It is also seen in the H groups that distant patterns, but nonetheless closer to H, are drawn towards the cluster (as shown in the dotted area of Fig. 2c). Recent results on much larger sets of patterns do not show departures from this situation.

In addition, if the input to the net is indeterminate, i.e. almost equally close in Hamming distance to both groups, the system either enters a recurring sequence of states or the resulting states cluster in an indeterminate area (Fig. 2b). Similar results are found with other sets of random connections, and it would be superfluous to present them here.

Finally, the importance of a controlled environment during the early stages of training must be emphasised. It was noted experimentally that, if, during the period of high teaching activity, random patterns are introduced into the training sequence, the ultimate performance of the net is greatly impaired. Separation of the two groups is largely lost and clustering becomes much less marked. Thus, as in human situations, it is difficult to dispense with the teacher altogether. Teaching of some kind must take place, albeit indirectly, in the sense that the range of patterns to which the system is exposed is controlled during its learning period.

Conclusion: The significance of the results lies in the fact that a randomly connected network *does* perform a clustering operation as one of its natural properties.

The reason for this is, fundamentally, that a random feedback network of logic elements inherently produces few and short cycles in its state space.[4] It is known that the states in these cycles and the state chains entering them are related in a Hamming-distance sense. The 'aging' exposure during the learning period brings about an association of the stable cycles with pattern classes, and the subsequent input patterns enter such cycles if they are related to them in the Hamming-distance sense. It is of engineering importance that a randomly connected network of adaptive elements should act in this 'artificially intelligent' way without the intervention of a human designer.

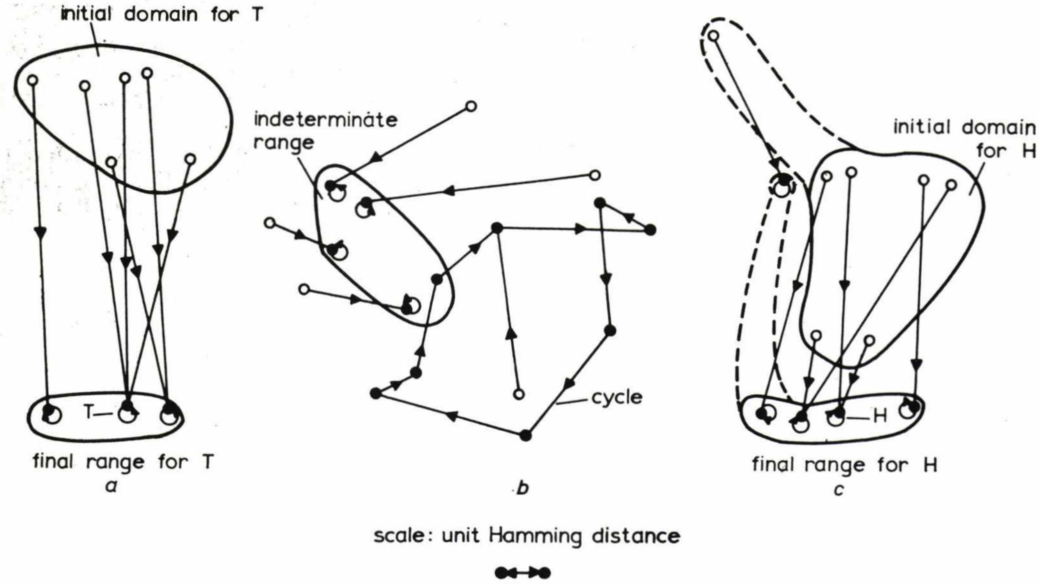


Fig. 2 *State-space dynamics*
● input
○ final state

Clearly, there is much work still to be done in an analysis of the process, and this is the objective of present research.

Functional description of a s.l.a.m.: A s.l.a.m. element is a random access memory used as a variable-logic/logic element. The memory address terminals ($a, b, c, \ldots$) are the inputs of the logic element, the single-bit readout terminal f is its output and the single-bit write terminal t is referred to as the 'teach' input. The 'teach clock' (h) input enables the writing mechanism.

As an example, the function of a 2-input element is given by

$$f = \phi_0\,\bar{a}\bar{b} + \phi_1\,\bar{a}b + \phi_2\,a\bar{b} + \phi_3\,ab$$

where

$$\phi_0' = \bar{a}\bar{b}th + \bar{h}\phi_0$$

$$\phi_1' = \bar{a}bth + \bar{h}\phi_1 \ldots$$

(Note that these are sequential equations, where ϕ_j is the 'present' state of a storage element and ϕ_j' the 'next' state.)

The authors wish to acknowledge the support of the UK Science Research Council.

M. C. FAIRHURST
I. ALEKSANDER

Electronics Laboratories
University of Kent
Canterbury, England

10th November 1971

References

1 ALEKSANDER, I., and MAMDANI, E. H.: 'Microcircuit learning nets: improved recognition by means of pattern feedback', *Electron. Lett.*, 1968, **4**, pp. 425–426

2 ALEKSANDER, I.: 'Some psychological properties of digital learning nets', *Int. J. Man-Mach. Stud.*, 1970, **2**, pp. 189–212

3 RUBIO, J. E.: 'The clustering and recognition of patterns', *Int. J. Contr.*, 1966, **4**, pp. 459–485

4 KAUFFMAN, S. A.: 'Metabolic stability and epigenesis in randomly constructed genetic nets', *J. Theor. Biol.*, 1969, **22**, pp. 437–467