Chapter 1

INTRODUCTION

1.1 Iterative Systems

Many problems of current interest in the field of information processing involve the presentation of data in the form of a simple sequence or a uniform array. The signals received over a teletype line constitute a time sequence; the numerical inputs to the accumulator of a digital computer constitute a one-dimensional array, or spatial sequence; and the light pattern projected upon the receptor cells of the eye constitutes a two-dimensional array. In these three examples, and in a large class of other problems, the manner in which the inputs are interpreted depends only upon the "pattern" of the signals, and is relatively independent of the exact time or location at which the signals appear. Thus a certain pattern of marks and spaces is interpreted as a teletype "A" regardless of the time at which it is received. Similarly, the operation of addition remains the same as the two numbers to be added are shifted with respect to the decimal point or the accumulator. Finally, the mind interprets the retinal image of a tree as a tree, regardless of where that image falls on the retina.

The fact that in many cases the kind of processing to be performed is unaffected by a translation of the input pattern suggests that the portion of the processing mechanism near any one input is similar in structure and operation to the portion near any other input. In the above examples this is indeed the case. The teletype receiver does not change its structure or mode of operation with time; each stage of the accumulator is identical; and the cells of the retina are, as far as we know, essentially alike in their structure and interconnection.

These examples, and others, suggest that an important class of information processing networks is that in which each network is composed of a number of identical subnetworks interconnected together to form a regular array. Networks constructed in this "iterative" form have several advantages over networks not having such a repeated structure. Being made up of many identical subnetworks, they are economical to manufacture and repair. They can be enlarged to accommodate more variables by simply adding more subnetworks; the existing portion of the network is unchanged. The design of an iterative network, which consists in specifying the circuitry of a single subnetwork, is usually simpler than that of an

equivalent unstructured network. Finally, an iterative network can often be converted from "parallel" to "sequential" operation without any major changes in design, while an unstructured network cannot.

It appears that many important and complex information processing problems will be implemented only when we are able to design, and build economically, large networks in iterative form. It is thus natural to ask, "What kinds of operations can iterative networks perform, how can they be analyzed, and how can an iterative network be designed to do a specific job?" The purpose of this book is to present a few preliminary answers to these very broad questions.

We shall be concerned only with systems in which the inputs and outputs occur at distinct points of space or time, rather than being distributed over the entire array. Furthermore, we shall restrict the operation of a system and its basic components to be discrete, rather than continuous. Thus the basic elements, and hence any network built from them, are to be logical, or switching, circuits. In principle, any iterative network of this type could be designed as a single switching circuit, but such a procedure would be exceedingly difficult for networks containing a large number of input variables. While it might be possible to state the input-output requirements concisely in English, a functional description of the same requirements would be hopelessly complex for all but the smallest networks. Even assuming that such a functional description could be obtained, there remains the complex task of reducing this description to an economical physical circuit. In particular, there is little chance that this design process would yield a network formed as an array of identical subnetworks, even when such an iterative solution is one of the most economical. If the advantages of the iterative solution are to be achieved, the designer must start off by assuming the general structure of the network and then determine the logical requirements that should be placed upon the individual subnetworks. While conventional switching theory is helpful in designing the specific circuitry of the subnetworks, it is of little help in deciding what the terminal behavior of these subnetworks should be. One of our aims is to gain some facility in prescribing this terminal behavior.

Although systems whose outputs are functions of a time sequence of input values are of great practical importance, we shall restrict our attention almost entirely to systems in which there is a fixed mapping of constant input patterns into constant output patterns. Any general treatment of systems whose terminal behavior is sequential in nature must wait until more is known about systems whose terminal behavior is combinational. The internal behavior of the systems will not be restricted, though, and may be sequential in nature. As we shall see, such sequential behavior has several advantages.

With this brief introduction to the kinds of problems we wish to consider, and reasons for considering them, we turn our attention to a more precise description of an iterative system. The next section sets forth the basic definitions that make possible a rigorous discussion of iterative systems, while the third section poses the questions and problems that will be treated in later chapters.

1.2 Definitions

Structure of an Iterative Network. An iterative network is one that is composed of a number of identical subnetworks, or cells, interconnected in a regular array. We shall restrict the cells to be logical networks, either combinational or sequential. One of each cell's inputs, called its primary input and denoted by the variable x, serves as an input to the entire network. In addition, there are a number of intercell leads that carry discrete signals between adjacent cells. In general, each cell will also have a primary output, denoted by the variable Z, which serves as one of the outputs of the entire network. With no loss of generality, we shall assume that the primary inputs and outputs are two-valued (with values 0 and 1).

There are many ways in which cells can be connected together to form an array. We shall consider only n-dimensional Euclidean arrays in which the location of any cell can be specified by n integer coordinates. Two such arrays are shown in Figure 1.1, where arrows

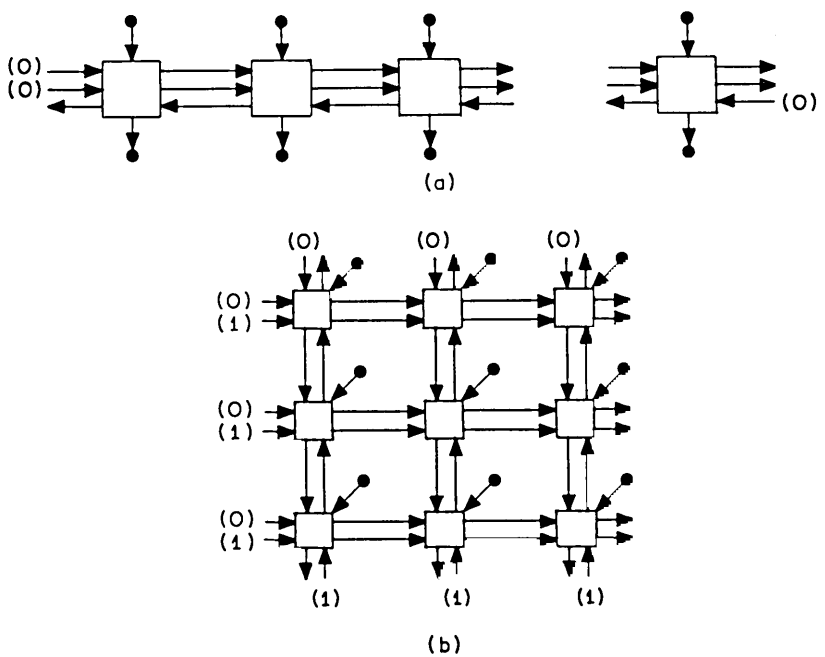


Fig. 1.1. Examples of iterative networks

are used to indicate the directions in which signals flow. In addition to specifying the logical structure of the cells that make up an iterative network, it is necessary to specify the signals that are to be applied to the intercell leads entering the edges of the networks. These specifications will be referred to as boundary conditions; the boundary signals are to be constant with time and identical for all the cells along any given boundary. In logical diagrams, the usual notation will be to indicate boundary conditions in parentheses, as in Figure 1.1.

For convenience of both analysis and synthesis, it will normally be assumed that an iterative network operates synchronously. Then if time instants are represented by integers, the outputs and internal state variables of any cell at time t are dependent only upon the inputs and internal state variables that were present in that cell at time t-1. At some points in the later chapters it will be possible to relax this restriction, but unless otherwise stated, we shall assume such a synchronism without indicating delay elements in the logical diagrams.

Note that no restriction has been placed upon the number of cells in an iterative network; we require only that the array of cells form a rectangle or hyper-rectangle. Thus the structure of an individual cell, together with a set of boundary conditions, is representative of an infinite number of iterative networks. The class of all the finite networks having a particular cell structure and boundary conditions is referred to as the iterative system defined by that cell structure and boundary conditions. Then the analysis problem for an iterative system consists in determining the behavior of an arbitrary network of the system in terms of the boundary conditions and the structure of an individual cell. Conversely, the synthesis problem consists in determining a satisfactory cell structure and boundary conditions in terms of the desired behavior of an arbitrary network.

In many cases we are not interested in obtaining a primary output from every cell in the network, but only in obtaining a single output from the entire network. This can most easily be done by ignoring the primary outputs of all cells except one, usually located at one end or corner of the network. Such a network will be referred to as a single-output network, and systems of such networks will be called single-output systems.

In other cases the primary output of each cell will be a function only of the position of that cell in the network, and not a function of the primary input values. A network or system in which the primary inputs do not influence the primary outputs is call autonomous.

We are now ready to make some definitions that will enable us to discuss the equilibrium and transient behavior of iterative systems.

Cell and Network States. The state of a cell of an iterative net-

work is that property described by the values of all the inputs to the cell, plus the internal state variables, if any. Thus, if the intercell leads of Figure 1.1a carry binary signals and the cells are combinational, each cell may assume sixteen different states.

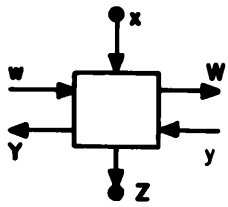
The state of an iterative network is that property described by the states of all its cells. Similarly, the state of a portion of a network is specified by the states of the cells in that portion. Thus a network of four of the cells of Figure 1.1a could have a total of 16^4 different combinations of cell states. When the boundary conditions are applied, this number is reduced to $8 \times 16^2 \times 4 = 8{,}192$ network states. Because of the assumption of synchronous operation, this particular network could be analyzed as a finite-state machine with 8,192 states. Although such an analysis is conceptually straightforward, it is clearly impractical. For practical purposes we seek a means of analysis that takes advantage of the repeated structure of the network, and one that is independent of the number of cells in the network.

Equilibrium and Transient Behavior. Fixing the values of the primary inputs of an iterative network will naturally reduce the number of states that the network can assume. As long as the input values remain constant, the network acts like an autonomous sequential machine, and can be represented by an appropriate state transition diagram.[13] If any network state is succeeded by itself in the transition diagram, we shall call it an equilibrium state for the particular pattern of primary input values chosen. It may happen that for some choice of primary input values no equilibrium states will exist, while for other choices one or more equilibrium states may exist.

If the transition diagram contains a closed loop involving two or more states, we shall say that the network has a (state) cycle for the chosen set of primary input values. This is necessarily the case when no equilibrium state exists, but may also occur when equilibrium states are present. Sometimes the states of a cycle will all produce the same pattern of primary output values; in other cases different primary output patterns will be produced. If a primary input combination results in a cycle in which different primary output patterns are produced, the network will be said to have an output cycle for that primary input combination. Clearly the existence of an output cycle implies the existence of a state cycle, but not the other way around.

If an iterative network has exactly one equilibrium state for each possible combination of primary input values, we shall call it a regular network. If an iterative system has the property that every one of its networks is regular, it will be called a regular system. If a network is free of state cycles for every possible choice of its primary input values, it will be called a stable network. Similarly, if every network in a system is stable, we shall call the system a

stable system. System stability is a very specialized property, but a very useful one. If a cell structure and boundary conditions define a stable system, then any network made up of these cells must ultimately reach an equilibrium state regardless of the initial state in which it is placed. Furthermore, as we shall see, all stable networks or systems of combinational cells are regular.



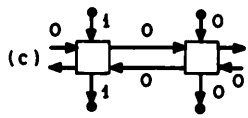
Boundary Conditions:

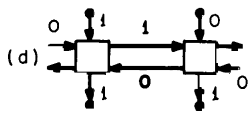
w = 0, y = 0

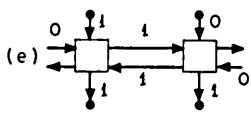
(a)

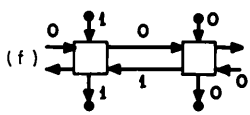
x	w	y	W	Y	Z
0	0	0	0	0	0
0	0	1	1	1	0
0	1	0	0	1	1
0	1	1	1	1	0
1	0	0	1	1	1
1	0	1	0	1	1
1	1	0	0	1	0
1	1	1	0	1	0

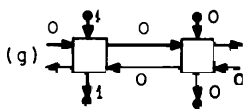
(b)



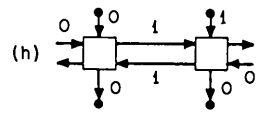




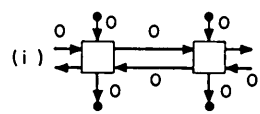


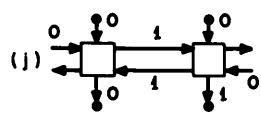


Network cycle



Unique equilibrium





Non-unique equilibrium

Fig. 1.2. Equilibrium states and cycles in iterative networks

At this point an example may be helpful. For reasons of simplicity we shall present a one-dimensional example, although the concepts described above apply equally well to networks of any number of dimensions. Figure 1.2a shows a combinational cell with two binary intercell leads, one carrying signals from left to right, the other carrying signals from right to left. Figure 1.2b summarizes the logical behavior of the cell in terms of a truth table. The boundary conditions require that w = 0 at the left-most cell of any network and that y = 0 at the right-most cell of any network. Figure 1.2c shows a network composed of two of these cells that is supplied with the designated boundary conditions and the constant primary input pattern (1, 0). The cell inputs presented by the intercell leads are initially assumed to be both 0. Then the left-hand cell must generate a 1 on its right output lead, while the right-hand cell must generate a 0 on its left output lead. Thus the cell inputs appearing at the next time instant are as shown in Figure 2.1d. Repeating this process, we find that the network must go through the states indicated in Figures 2.1c-g at successive time instants. Since the state of the network at the fifth time instant is identical to that at the first, the network exhibits a cycle for this particular primary input combination. Furthermore, all four possible network states appear in the cycle, so that no equilibrium state exists for this input combination.

When the primary inputs are (0, 1), the example network has a unique equilibrium state, as shown in Figure 1.2h. That the equilibrium is unique can be verified by showing that each of the three other possible network states eventually leads to this equilibrium. Finally, if the primary inputs are both 0, the network has two equilibrium states, shown in Figures 1.2i and 1.2j. Note that these equilibrium states are essentially different in that they produce different primary output patterns. In conclusion, we note that the system described in Figure 1.2b is neither stable nor regular, since it contains a network that exhibits a cycle for some primary input pattern, as well as one that lacks an equilibrium state for some primary input pattern.

Equivalence. Suppose that two regular iterative networks have different cell structures and boundary conditions, but are composed of the same number and arrangement of cells. Now apply identical primary input patterns to the two networks and examine the primary output patterns produced when the networks are in equilibrium. If the equilibrium primary output patterns of the two networks match exactly for all possible input patterns, the networks will be said to be equivalent. Two regular systems are equivalent if and only if each network of one is equivalent to the corresponding network of the other. If single-output networks or systems are being considered, only those primary outputs that actually represent network outputs are to be compared.

Composition. One operation that we shall make frequent use of in later chapters is the combination of two networks of the same size into one composite network. One possible way to combine two networks of the same number and arrangement of cells is simply to connect their corresponding primary inputs together. In this case we shall say that they have been connected in parallel. Each cell of the parallel combination contains as its component parts the cell structures of the two original networks. Figure 1.3 illustrates this process. In this figure, as in others to follow, heavy arrows are used to represent bundles of binary leads. The primary outputs of the composite network may be specified as some function of the primary outputs of the component networks, or more generally as some function of the states of the composite network.

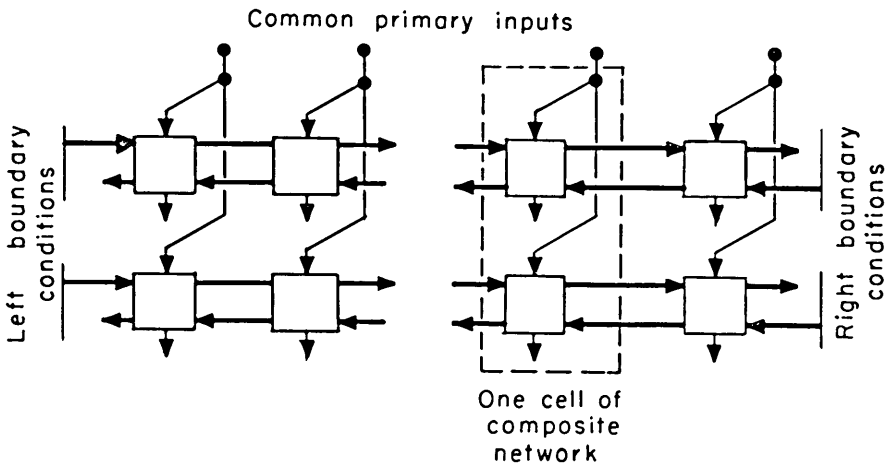


Fig. 1.3. Parallel combination of iterative networks

If every network of one system is placed in parallel with the corresponding network of a second system, the resulting system is called the parallel combination of the two original, or component, systems. The intercell signals of the composite system, which are referred to as composite signals, consist of ordered pairs of signals from the component systems. In general, not all possible pairs of this type will occur in the composite system. The composite signals that are actually needed may usually be determined by starting at the boundary conditions and working inwards, step by step. The details of this process will be described in Chapter 2.

1.3 Problems for Consideration

Our ultimate goal is to be able to design iterative information processing systems that realize any specified mapping of primary input patterns into primary output patterns. The terminal behavior of such a system is to be "memoryless" in the sense that the time sequence of inputs preceding the appearance of a given input pattern has no effect upon the output pattern to be produced by that particular input. This does not imply that the internal behavior of the system must be memoryless; important advantages are often to be gained through the use of sequential cell structures. Although it is not always possible, one simple way to design these systems is to require that each network in the system have some equilibrium state for every possible primary input pattern. To facilitate this approach, it would be desirable to have a test that could determine whether or not every network of a given system possessed an equilibrium state for each possible primary input pattern. A further refinement would be a test for deciding whether a given system is regular, i.e., whether every network in the system has a unique equilibrium for each primary input pattern. Another useful analysis tool would be a test that could determine whether or not two regular systems were equivalent. Thus we should like to have general tests that could be applied to a description of the cell structure and boundary conditions of an arbitrary system in order to provide answers to a few fundamental questions about the equilibrium, or steady-state, behavior of the system. Unfortunately, such tests do not exist for all classes of iterative systems. Chapter 2 defines certain classes of systems for which these tests do exist, and describes the tests in detail. Chapter 3 is devoted to proving that corresponding tests for other classes of systems do not exist.

In Chapter 4, we consider the analysis of the transient behavior of iterative systems, and in particular, the possibility of formulating a general procedure for testing the stability of an arbitrary system. Here the results are even more discouraging, since general stability tests do not exist for any class of systems containing feedback loops.

Having established that even the simplest equilibrium and transient questions cannot be answered in general for many classes of iterative systems, and that consequently a general analysis procedure does not exist, we turn our attention to the functional capabilities of various types of systems. Chapter 5 discusses the effects of dimensionality, cell memory, and directions of signal flow upon the capabilities of iterative systems. The results obtained on the analytical aspects of iterative systems are summarized in Chapter 6.

Chapter 7 presents techniques applicable to the synthesis of

the simplest class of systems, namely one-dimensional systems with one direction of signal flow. Techniques for designing networks without internal cell memory are first reviewed, and then extended to networks with cell memory. In Chapter 8, means of dealing with problems of stability, premature outputs, and asynchronous operation are discussed. Chapter 9 attempts to expand the techniques of Chapter 7 to apply to one-dimensional networks with two directions of signal flow, and to two-dimensional networks. Finally, extensions of the familiar minimization techniques to apply to more complex systems are discussed in Chapter 10. Chapter 11 concludes with a summary and discussion of important unsolved problems.